3

CEKIT S.A.

Curso práctico sobre

MICROCONTROLADORES

Teoría, Programación, Diseño, Prácticas y Proyectos completos

Teoría

Los microprocesadores
Evolución histórica

Programación

Conceptos básicos

Lenguajes de programación

Proyectos

Introducción

Pasos a seguir para elaborar un proyecto

Argentina: $ 3.90
Colombia: $ 5.500
Chile: $ 1.450
México: $ 27
Venezuela: Bs. 2.000

Tomo 3
- Proyectos -

Curso práctico sobre

MICROCONTROLADORES

Teoría, Programación, Diseño, Prácticas y Proyectos completos

El **Curso práctico sobre MICROCONTROLADORES** de CEKIT S. A. se publica en forma de 40 fascículos de aparición semanal, encuadernables en 3 volúmenes. Cada fascículo consta de 4 páginas de cubiertas y 20 páginas de contenido a todo color. De estas últimas, 8 están dedicadas a la sección de **TEORIA**, 4 a **PROGRAMACIÓN** y 8 a la sección de **PROYECTOS.**

Las páginas de cada sección son encuadernables por separado. Por esta razón, al formar los volúmenes, debe cuidarse de desprender previamente, de todos los fascículos, las 8 páginas centrales de **PROYECTOS**, las 4 de **PROGRAMACIÓN** y por último las 8 páginas de **TEORÍA**. Durante la publicación de los fascículos se pondrán a la venta las tapas de los 3 volúmenes de la obra para su encuadernación.

CEKIT S.A. garantiza la publicación de la totalidad de la obra, el suministro de la tapas necesarias para su encuadernación y el servicio de números atrasados hasta un año después de terminada la circulación del último fascículo. También garantiza la fidelidad de la información teórico-práctica suministrada y el correcto funcionamiento de los programas incluidos como ejemplos de la obra.

Motorola, Microchip, PIC, Intel, Atmel, Parallax, Zilog, Phillips, son marcas registradas de las diferentes empresas en los Estados Unidos y en otros paises. Otros productos y nombres de compañías mencionadas en esta obra pueden ser marcas registradas de sus propietarios.

Este curso ha sido elaborado según el plan del editor y de sus autores, y bajo su responsabilidad, por los siguientes integrantes del departamento técnico de CEKIT S.A.

Distribución Internacional

Argentina: Importador: Editorial Conosur S.A. Av. Belgrano 355 10º. Tel: 4342-9029 Fax: 4342-9025 e-mail: gconosur@speedy.com.ar Dist. Capital: VACCARO SANCHES Y CIA. - Moreno 794, 9º(1092) Buenos Aires. Dist. Interior: DISTRIBUIDORA BERTRAN S.A.C. - Av. Vélez Sarsfield 1950(1285) Buenos Aires. - Bolivia: AGENCIA MODERNA LTDA. - Chile: DISTRIBUIDORA ALFA S.A. - Colombia: DISTRIBUIDORAS UNIDAS. - Venezuela: DISTRIBUIDORA CONTINENTAL.- México: CONSORCIO SAYROLS - Ecuador: DISTRIBUIDORA ANDES. - Panamá: PANAMEX S.A. - Perú: DISTRIBUIDORA BOLIVARIANA S.A. - Paraguay: SELECCIONES S.A.C. - Uruguay:DISTRIBUIDORA CAREAGA

Suscripciones (no válidas en Argentina), números atrasados y consultas técnicas en Colombia

CEKIT S.A.
Calle 22 Nº 8 - 22, Piso 2, A.A. 194 Pereira(Colombia)
o a nuestro correo electrónico:
ecekit@col2.telecom.com.co

ISBN (Fascículo 8):958-657-510-1
ISBN (Volumen 1):958-657-498-9
ISBN (Obra completa): 958-657-497-0

Calle 22 Nº 8 - 22, Piso 2,
Teléfonos
(6) 3348179- 3348187-
3348189 - 3348193
Fax: (6) 3348020
Pereira-Colombia
Carrera 13 Nº 38 - 43, Piso 1.
Teléfonos: 2873086 -
2873420 - 2877318
Bogotá - Colombia

e-mail: ecekit@col2.telecom.com.co
http://www.cekit.com.co

Gerente General:
Jorge Hernán Alvarez
Gerente Comercial:
Marcelo Alvarez H.
Director Editorial:
Manuel Felipe González G.
Circulación Internacional:
Humberto Real Blanco.

Autores

JUAN ANDRÉS CASTAÑO W.
MANUEL FELIPE GONZÁLEZ G.
GILBERTO VARGAS C.
JUAN DIEGO HERNÁNDEZ F.

Proyectos

JOHN JAIME ROBBY G.

Diseño carátula: Luz Angela Plata B.
Diseño: Gloria Marcela Gómez G.
Diagramación: Héctor Hugo Jiménez

© CEKIT S.A. 2001
Pereira - Colombia

Todos los derechos reservados. Prohibida su reproducción parcial o total por cualquier medio sin permiso escrito del editor.

EDICIÓN CONO SUR
Internacional Graphics & Printing Co.
2600 Douglas Road, Suite 406
Coral Gables, Florida 33134, U.S.A.

Impreso y encuadernado por:
RR Donnelley
América Latina
Ruta Panamericana Km. 38.700
Buenos Aires Argentina
Impreso en Argentina
Junio 2002

Curso práctico sobre

MICROCONTROLADORES

Teoría, Programación, Diseño, Prácticas y Proyectos completos

Estudie, con el método de CEKIT de **"Aprender haciendo"**, la tecnología de los Microcontroladores, una de las que más han cambiado el mundo en los últimos años y gane mucho dinero. Estos pequeños cerebros electrónicos se encuentran en todas partes: automóviles, computadoras, electrodomésticos, alarmas, juegos de luces, instrumentos, equipos de comunicaciones, etc.

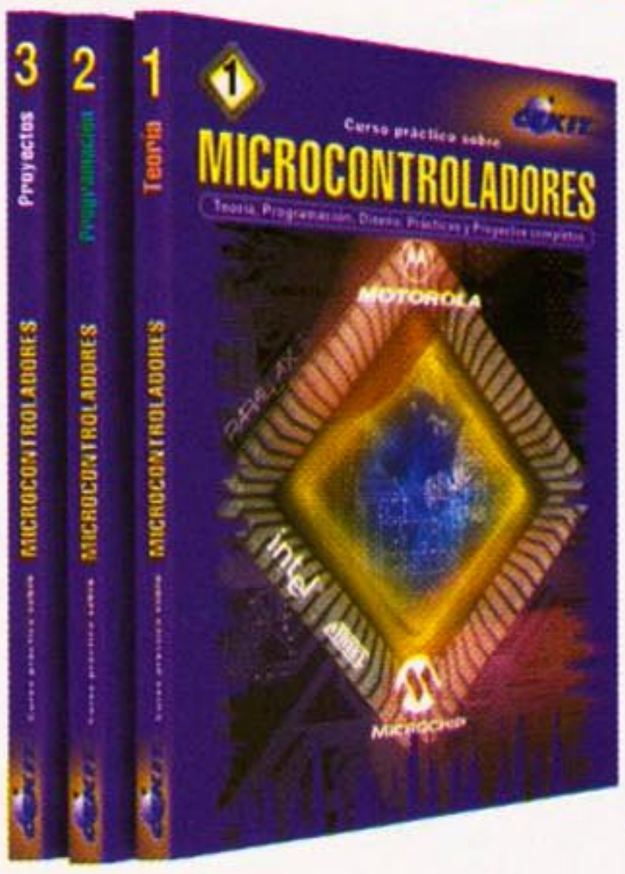
En este curso incluimos los principios teóricos, los fundamentos de su programación, una gran cantidad de actividades prácticas y proyectos completos para ensamblar.

Los microcontroladores son los elementos básicos y fundamentales de una nueva revolución, la interconexión de todo con todo en una red mundial de información y control. Esto permitirá la comunicación de hechos y datos entre personas y objetos formando la red del conocimiento, base de la economía del nuevo milenio **Capacítese y forme parte activa de la nueva economía.**

Libere su creatividad

Las personas con buenos conocimientos de esta tecnología, tendrán muchas oportunidades laborales o podrán fabricar sus propios aparatos en forma industrial, lo que puede ser la base para una empresa muy rentable.

Como está estructurado el curso



El **Curso práctico sobre Microcontoladores** de **CEKIT S.A.** se publicará en 40 fascículos con tres secciones cada uno, y al final, se agrupará en tres tomos de pasta dura con cada uno de los temas principales. Para su fácil identificación, cada una de las secciones tiene un color diferente. La distribución de los tomos es la siguiente:

Tomo 1: Teoría	Tomo 2: Programación	Tomo 3: Proyectos
320 páginas (1 a 320)	160 páginas (1 a 160)	320 páginas (1 a 320)

Sección de Teoría

En la sección de teoría, se estudian los fundamentos teóricos y funcionales de los microcontroladores. Esto se refiere a la estructura o arquitectura interna de los mismos, así como a la función de cada uno de los terminales o pines de los diferentes dispositivos tratados en el curso cuyo conocimiento es necesario para diseñar circuitos con estos componentes.

Se empieza con un repaso sobre el tema de los microprocesadores, los cuales dieron origen a los microcontroladores, y se desarrollan durante las lecciones, los diferentes aspectos técnicos de esta tecnología.

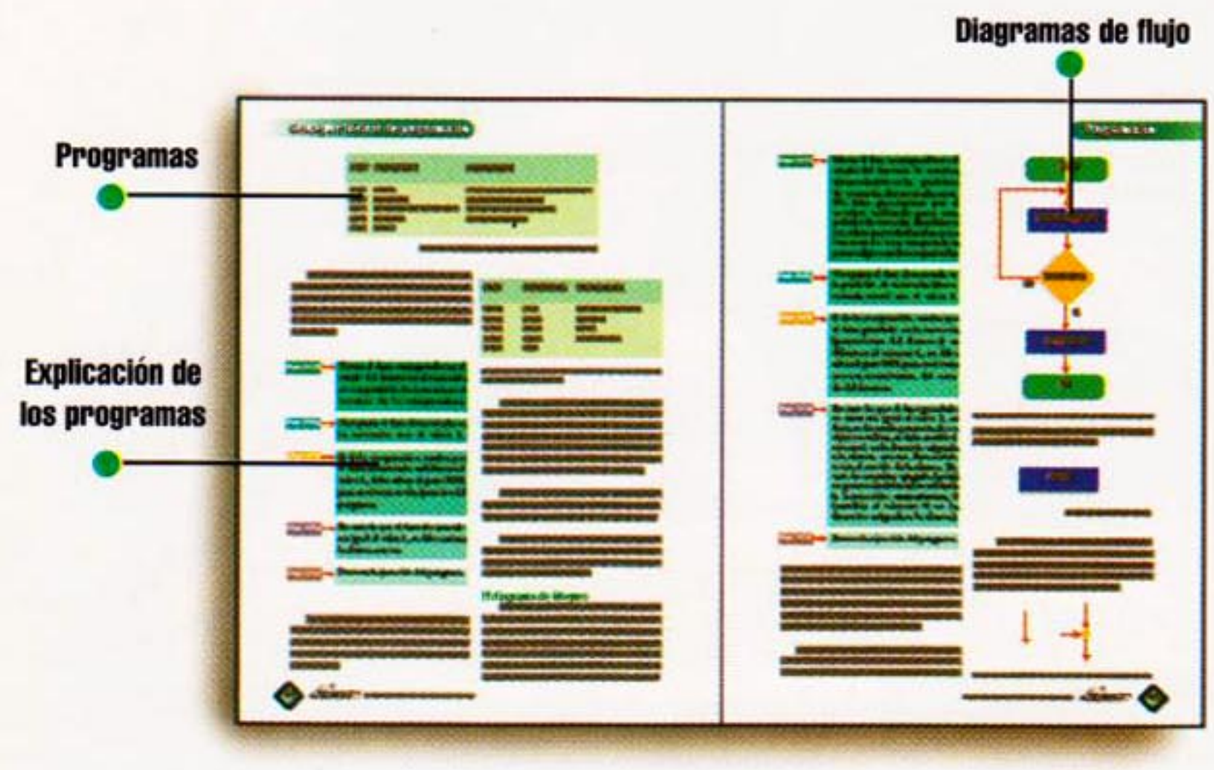
Los siguientes son los principales temas tratados en esta sección:

- Introducción a los microprocesadores
- Introduccion a los microcontroladores
- Diferencias entre un microprocesador y un microcontrolador
- Principales familias de microcontroladores
- Estuctura básica de los sistemas con microcontrolador
- (Motorola y PIC de Microchip)
- Arquitecturas de microcontroladores
- La CPU y la ALU
- Los registros
- Organización de la memoria
- La EPROM de datos
- Las interrupciones
- Los puertos de entrada/salida (I/O)
- Los temporizadores (*Timers*)
- Las opciones Comparación/Captura/PWM
- El puerto serial síncrono SSP, bssp
- La USART
- Las referencias de voltaje
- Los comparadores
- Los conversores A/D
- Modos de bajo consumo (*sleep* y *watchdog*)
- Bits de configuracion
- Programación en el circuito (*in circuit*)
- El conjunto (*set*) de instrucciones
- Las comunicaciones seriales
- Otros microcontroladores
- Parallax (Basic Stamp)
- Intel
- Atmel

Sección de Programación

Todo aparato o proyecto con microcontroladores incluye, además de los diferentes circuitos, uno o varios programas que se deben encribir en el lenguaje apropiado según el tipo de microcontrolador utilizado. Este curso le proporcionará los conocimientos necesarios para que adquiera la habilidad de programarlos.

Los principales temas tratados son:

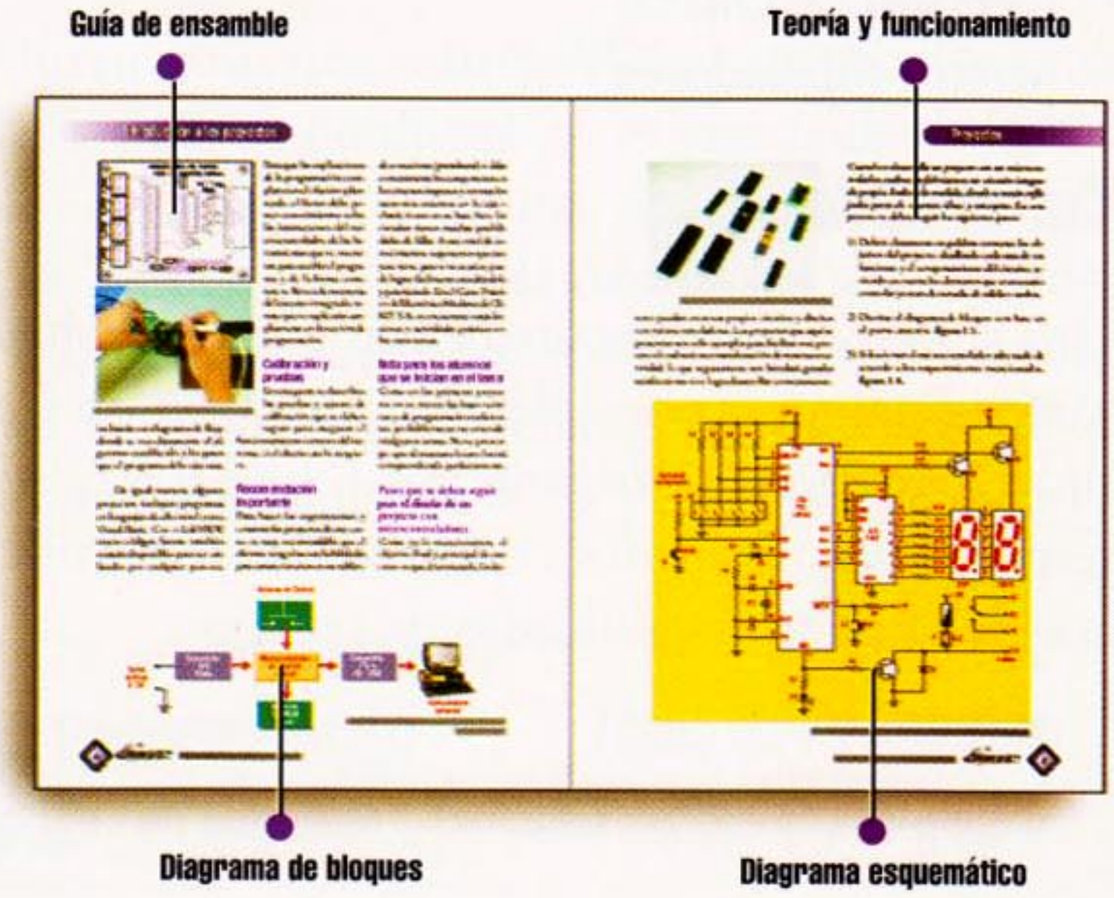


- Programación de un microcontrolador
- El ciclo de desarrollo de un programa
- Herramientas de *software*
- Programación en modo inmediato
- Programación orientada a registros
- Control de flujo de un programa
- Puertos de entrada/salida
- Subrutinas
- Interrupciones
- Programación en lenguaje estructurado
- Temporizadores
- Programación serial
- Programando los conversores A/D y D/A
- Programación en lenguaje C
- Tópicos avanzados de programación

Sección de proyectos

- Indicador binario de 4 bits
- Alarma para el hogar y el automóvil
- Reloj digital y calendario
- Secuenciador de luces programable
- Cerradura codificada
- Activación de aparatos vía telefónica
- Temporizador universal
- Aviso luminoso con diodos LED
- Luces rítmicas
- Convertidor paralelo-serial
- Punta lógica
- Temporizador miniatura
- Frecuencímetro
- Generador de señales
- Programador de memorias
- Adquisicion de datos en 16 bits
- Minirobot con microcontrolador
- Adquisición de datos por el puerto serial
- Tarjeta I/O para el puerto serial
- y muchos otros...

En esta importante sección del curso, se entrega una amplia información con la cual el alumno podrá ensamblar una buena cantidad de proyectos. Para cada uno de ellos se incluye la teoría de funcionamiento, el diagrama esquemático, la lista de materiales, las instrucciones para el ensamblaje y el listado del programa.



A quienes va dirigido el curso

El tema de los microcontroladores es de gran utilidad para un grupo muy amplio de personas que requieran conocer esta tecnología por su gran cantidad de aplicaciones.

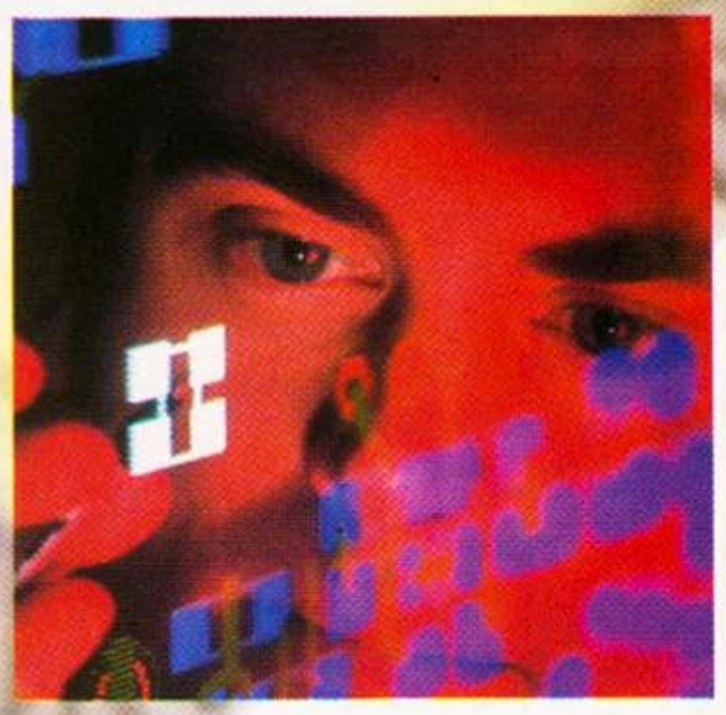
Los principales grupos de usuarios son:

Estudiantes y profesionales de Ingeniería y Tecnologías:

- Electrónica
- Eléctrica
- Mecánica
- Sistemas
- Bioingeniería
- Mecatrónica
- Automatización
- Estudiantes de colegios técnicos con énfasis en electrónica

Técnicos en:

- Electrónica industrial
- Automatización
- Electrónicos en general
- Alumnos de los cursos de **CEKIT S.A.**
- Estudiantes y profesionales en FISICA, para sus necesidades de instrumentación



Autodidactas o aficionados con conocimientos en:

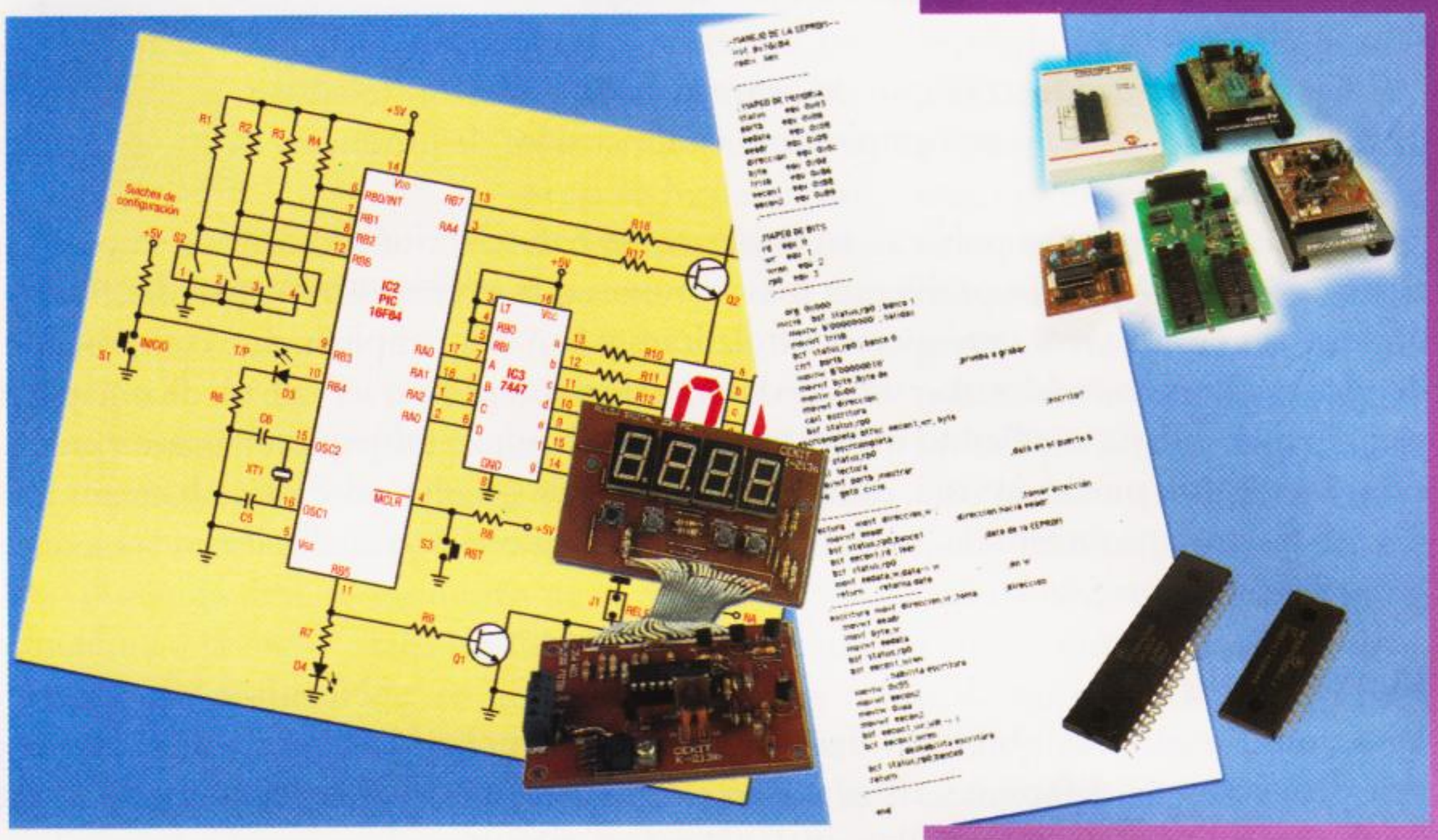
- Electrónica Digital
- Microprocesadores

Certificado

Una vez terminado el curso y aprobada la evaluación que se publicará al finalizar la obra, el alumno recibirá un Certificado de Aptitud Ocupacional otorgado por CEKIT S.A. y aprobado por las respectivas autoridades educativas. Este certificado ler permitirá acceder a puestos calificados de trabajo.

Otra obra con la calidad editorial y didáctica de CEKIT S.A.®
Capacitamos para vivir mejor

INTRODUCCIÓN



PROYECTOS CON MICROCONTROLADORES

En esta sección tendremos una serie de proyectos completos con diferentes tipos de microcontroladores con el fin de practicar y comprobar la teoría expuesta, y así cumplir con los objetivos didácticos esperados. En todos los cursos de electrónica, a cualquier nivel, si no se practica es muy difícil aprender algo. Por lo tanto, recomendamos que los alumnos construyan, ojalá, todos los proyectos que aquí se presentan lo que les garantizará el verdadero aprendizaje de esta tecnología de tanta actualidad y aplicaciones.

Podríamos decir que ésta es la sección más importante del curso ya que aquí se ponen en práctica todos los conocimientos adquiridos en las demás secciones, pero teniendo siempre presente que un proyecto con microcontroladores, sin las bases teóricas, no tiene mucho valor. Una vez que el lector haya terminado el curso, solamente podrá decir que habrá aprendido si es capaz de realizar sus propios proyectos partiendo de un problema o una necesidad; logrando además una solución original la cual puede convertirse en un producto final de amplia aceptación en el mercado lo cual le traería beneficios económicos tanto a nivel personal como local, regional y nacional.

Un buen diseñador y ejecutor de proyectos con microcontroladores, siempre tendrá oportunidades de trabajo, ya sea como empleado o como creador de productos electrónicos innovadores.

Los proyectos que se publican en este curso se han seleccionado como ejemplos muy representativos de soluciones a los problemas más comunes que se encuentran en el diseño de todo tipo de dispositivos electrónicos con microcontroladores y han sido ampliamente comprobados. Cada uno de ellos puede ser considerado como una aplicación completa o ser la base de un aparato más complejo que se obtiene añadiendo más funciones para cumplir los requerimientos exactos de esa solución. Desde otro punto de vista, estos se pueden tomar simplemente como la mejor forma de visualizar el desarrollo concreto de los conceptos teóricos que se han analizado en las secciones de teoría general y programación.

Metodología

Para cumplir con los objetivos de esta sección, se incluyen proyectos de diferentes tipos como juegos de luces, temporizadores, alarmas, relojes, instrumentos, circuitos de control, circuitos que se conectan a los puertos serial o paralelo de una computadora tipo PC, circuitos de comunicaciones, etc. **figura 1.1**. Cada uno de estos proyectos incluye:

Nombre, objetivo e importancia del proyecto

En este punto se destacan sus aplicaciones, su posibilidad de formar parte en otros proyectos más complejos y las alternativas de expansión o modificación para obtener variantes del diseño original.

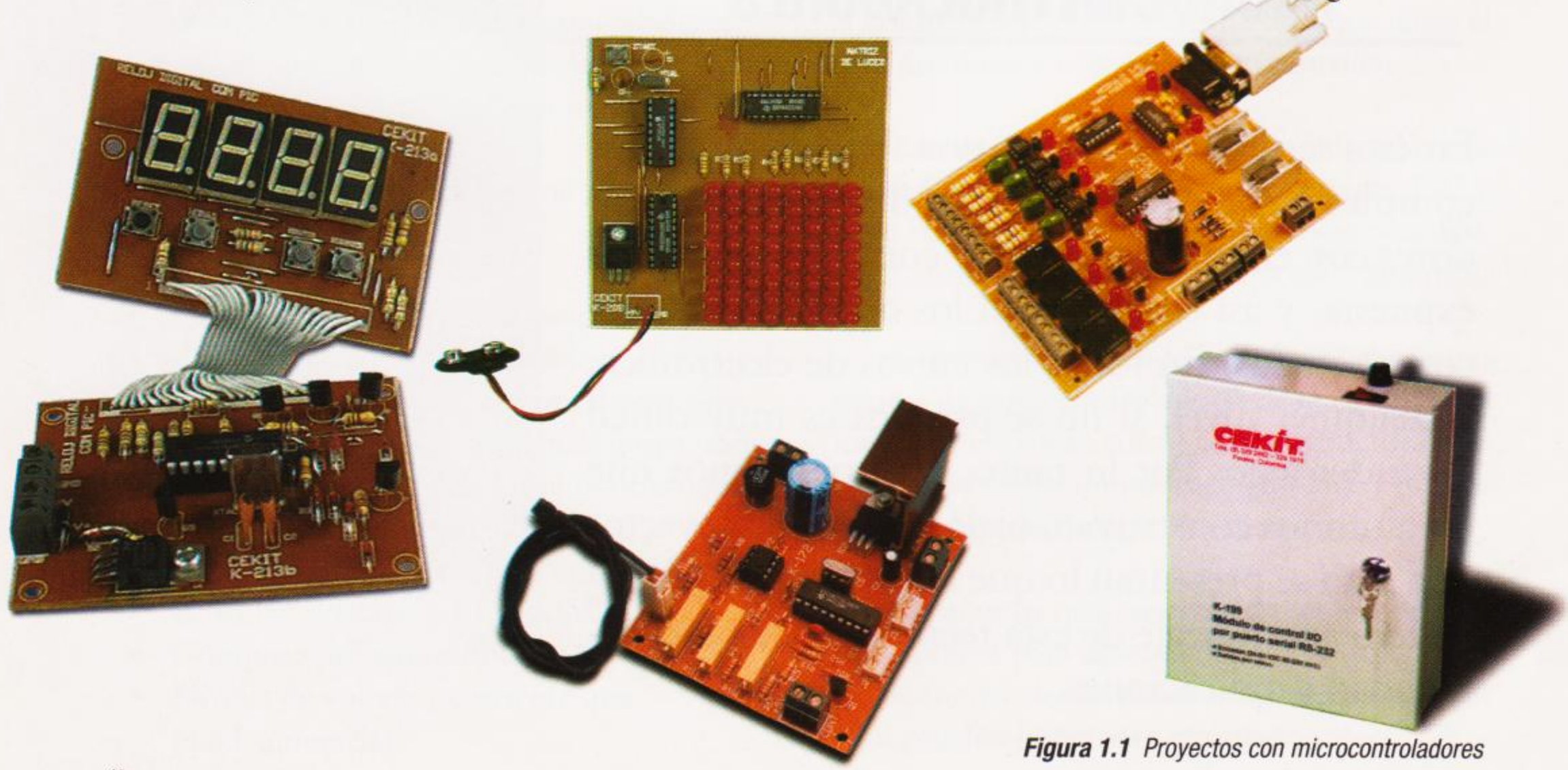


Figura 1.1 *Proyectos con microcontroladores*

Explicación del diseño

Es la explicación concisa y detallada de cada una de las partes que forman el proyecto. Para hacer esto de la forma más clara y didáctica posible, se acudirá a diagramas de bloques, simulaciones por computadora y descripción paso a paso de cada una de las etapas que forman el sistema. También se incluirán diagramas de tiempos de las señales digitales que se manejen, en donde la aplicación lo amerite.

Esta explicación incluye el diagrama esquemático o plano en el cual se muestran todos los componentes del circuito y la forma como se interconectan entre ellos.

Componentes complementarios

Cada proyecto de este tipo posee, además del microcontrolador, una serie de componentes adicionales que realizan funciones de entrada como recibir y acondicionar señales de diferentes tipos (sonoras, luminosas, de voltaje, de corriente, etc.); detección de órdenes como en el caso de interruptores, pulsadores y teclados, entre otros; conversores análogo/digitales (A/D); circuitos de reloj en tiempo real; circuitos de comunicaciones con otros dispositivos similares ya sea hacia un sólo elemento o a una red, etc., **figura 1.2.**

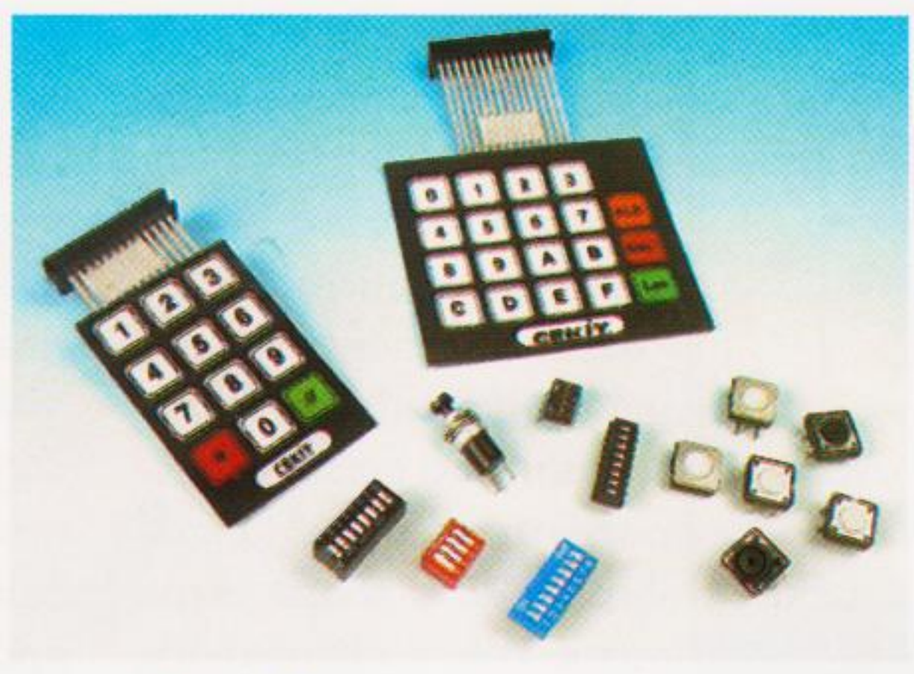


Figura 1.2 *Componentes con funciones de entrada*

En cuanto a los dispositivos de salida, se pueden tener otros circuitos integrados ya sean de uso general o especiales, indicadores luminosos como diodos LED, pantallas (*displays*) o indicadores de siete segmentos, pantallas de cristal líquido (LCD), indicadores sonoros, relés electromagnéticos o de estado sólido, optoacopladores, transistores, triacs, solenoides, motores, etc., **figura 1.3.** En cada proyecto se explicarán las características y el funcionamiento de estos componentes.

Lista de materiales

Todos los proyectos tienen una lista completa de los materiales que los conforman para facilitar su adquisición en el mercado. Cada componente se enumera con su respectiva etiqueta y valor o referencia que lo identifica dentro del diagrama esquemático.

Instrucciones de ensamblaje

En este curso se hará un especial énfasis en los cuidados e instrucciones para armar cada uno de los proyectos, para lo cual se mostrará gráficamente la evolución de este proceso. Con este fin también se incluirá una guía para la correcta ubicación de los componentes y la conexión de los accesorios externos que forman parte del circuito o aparato, **figura 1.4.**

Explicación de los programas (*software*)

Aunque en la sección de programación se explican los conceptos generales para el diseño de los programas, en cada proyecto se mostrará su programa completo, debidamente documentado, si el espacio lo permite. Si es muy extenso, éste se publicará, igual que los demás, en la página web de CEKIT S.A. (**www.cekit.com.co**) en una sección especial creada para este curso. Todos los programas

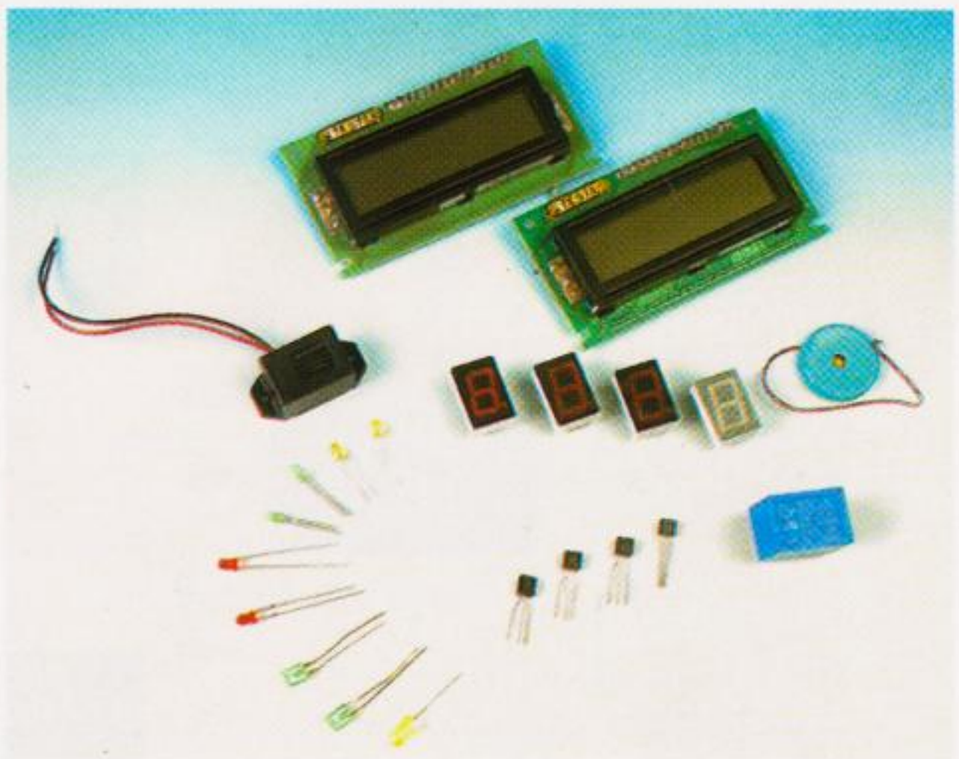


Figura 1.3 *Componentes con funciones de salida*

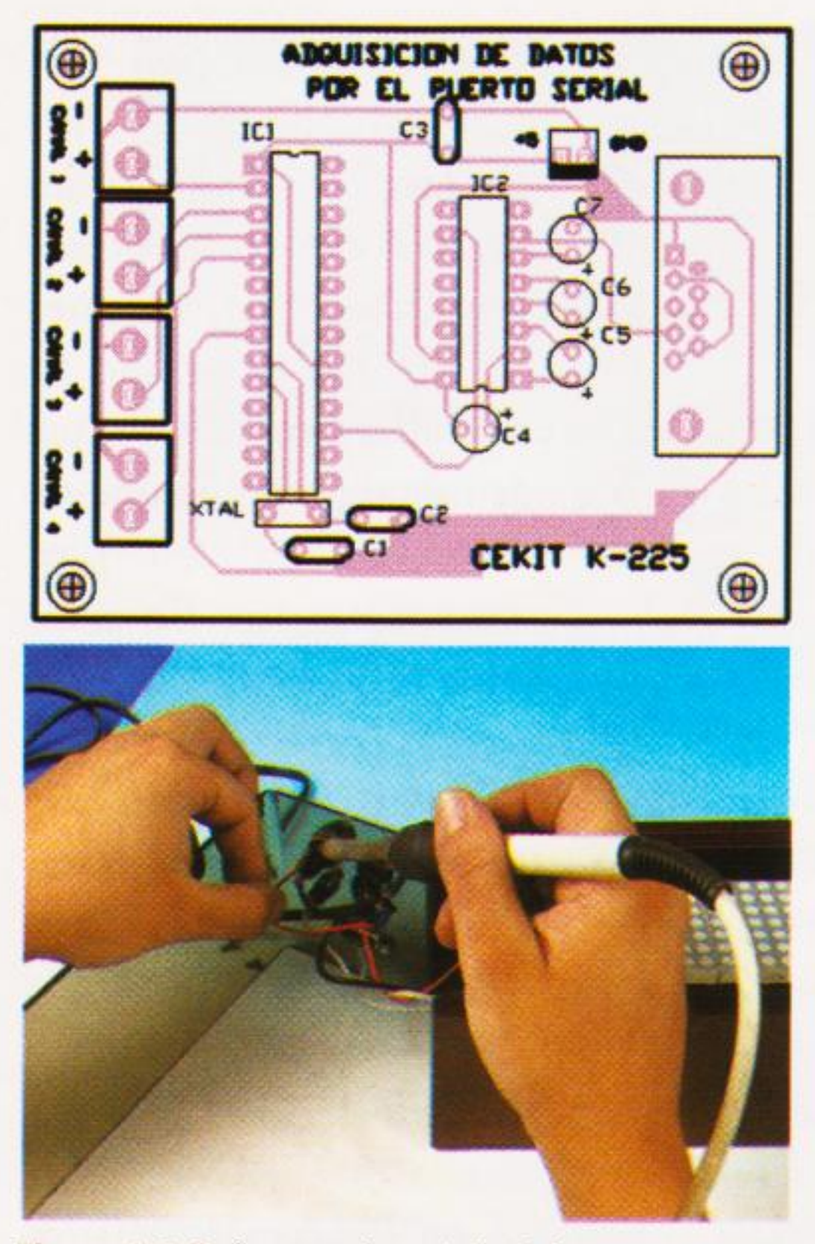


Figura 1.4 *Guía para el montaje de los componentes*

incluirán un diagrama de flujo donde se vea claramente el algoritmo establecido y los pasos que el programa debe ejecutar.

De igual manera, algunos proyectos incluyen programas en lenguajes de alto nivel como Visual Basic, C++ o LabVIEW cuyos códigos fuente también estarán disponibles para ser utilizados por cualquier persona.

Para que las explicaciones de la programación cumplan con el objetivo planteado, el lector debe poseer conocimientos sobre las instrucciones del microcontrolador, de las herramientas que se necesitan para escribir el programa y de la forma como éste se lleva a la memoria del circuito integrado; temas que se explicarán ampliamente en la sección de programación.

Calibración y pruebas

En esta parte se describen las pruebas y ajustes de calibración que se deben seguir para asegurar el funcionamiento correcto del sistema, si el diseño así lo requiere.

Recomendación importante

Para hacer los experimentos y construir los proyectos de este curso, es muy recomendable que el alumno tenga buenas habilidades para armar circuitos en un tablero de conexiones (*protoboard*), soldar correctamente los componentes en los circutios impresos y montar los accesorios externos en la caja o chasis; si esto no se hace bien, los circuitos tienen muchas posibilidades de fallar. A este nivel de conocimientos, suponemos que esto ya se tiene, pero si no es así, se puede lograr fácilmente estudiándolo y practicando. En el Curso Práctico de Electrónica Moderna de CEKIT S.A., se encuentran varias lecciones y actividades prácticas sobre estos temas.

Nota para los alumnos que se inician en el tema

Como en los primeros proyectos no se tienen las bases teóricas y de programación suficientes, probablemente no entenderá algunos temas. No se preocupe, que al avanzar el curso los irá comprendiendo perfectamente.

Pasos que se deben seguir para el diseño de un proyecto con microcontroladores

Como ya lo mencionamos, el objetivo final y principal de este curso es que al terminarlo, los lec-

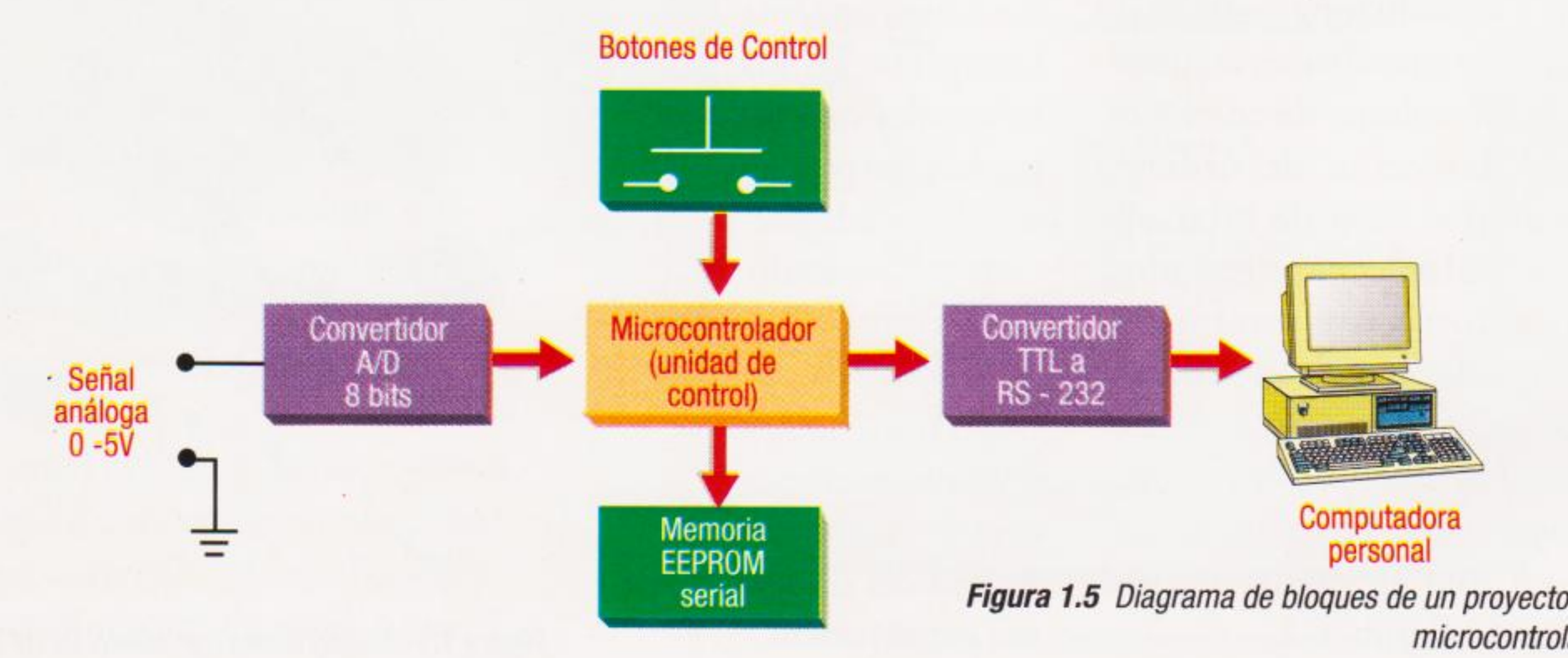


Figura 1.5 *Diagrama de bloques de un proyecto con microcontrolador*

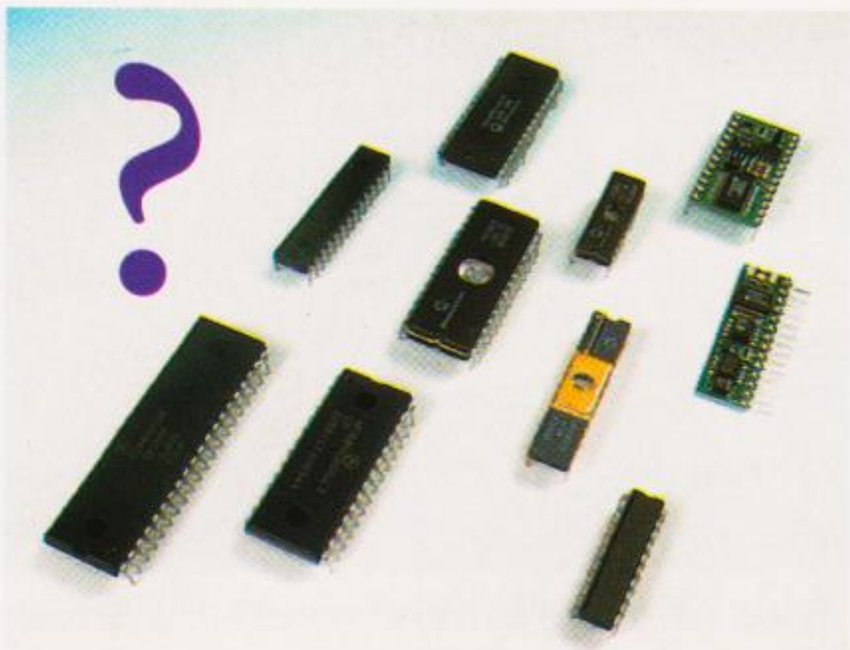


Figura 1.6 *Diferentes tipos de microcontroladores*

tores puedan crear sus propios circuitos y diseños con microcontroladores. Los proyectos que aquí se presentan son sólo ejemplos para facilitar este proceso el cual será una manifestación de nuestra creatividad, lo que seguramente nos brindará grandes satisfacciones si se logra desarrollar correctamente.

Cuando se diseña un proyecto con un microcontrolador, realmente fabricamos un circuito integrado propio, hecho a la medida, donde se verán reflejadas parte de nuestras ideas y conceptos. En este proceso se deben seguir los siguientes pasos:

1) Definir claramente en palabras comunes los objetivos del proyecto, detallando cada una de sus funciones y el comportamiento del circuito, teniendo en cuenta los elementos que es necesario controlar ya sean de entrada, de salida o ambos.

2) Diseñar el diagrama de bloques con base en el punto anterior, **figura 1.5**.

3) Seleccionar el microcontrolador adecuado de acuerdo a los requerimientos mencionados, **figura 1.6**.

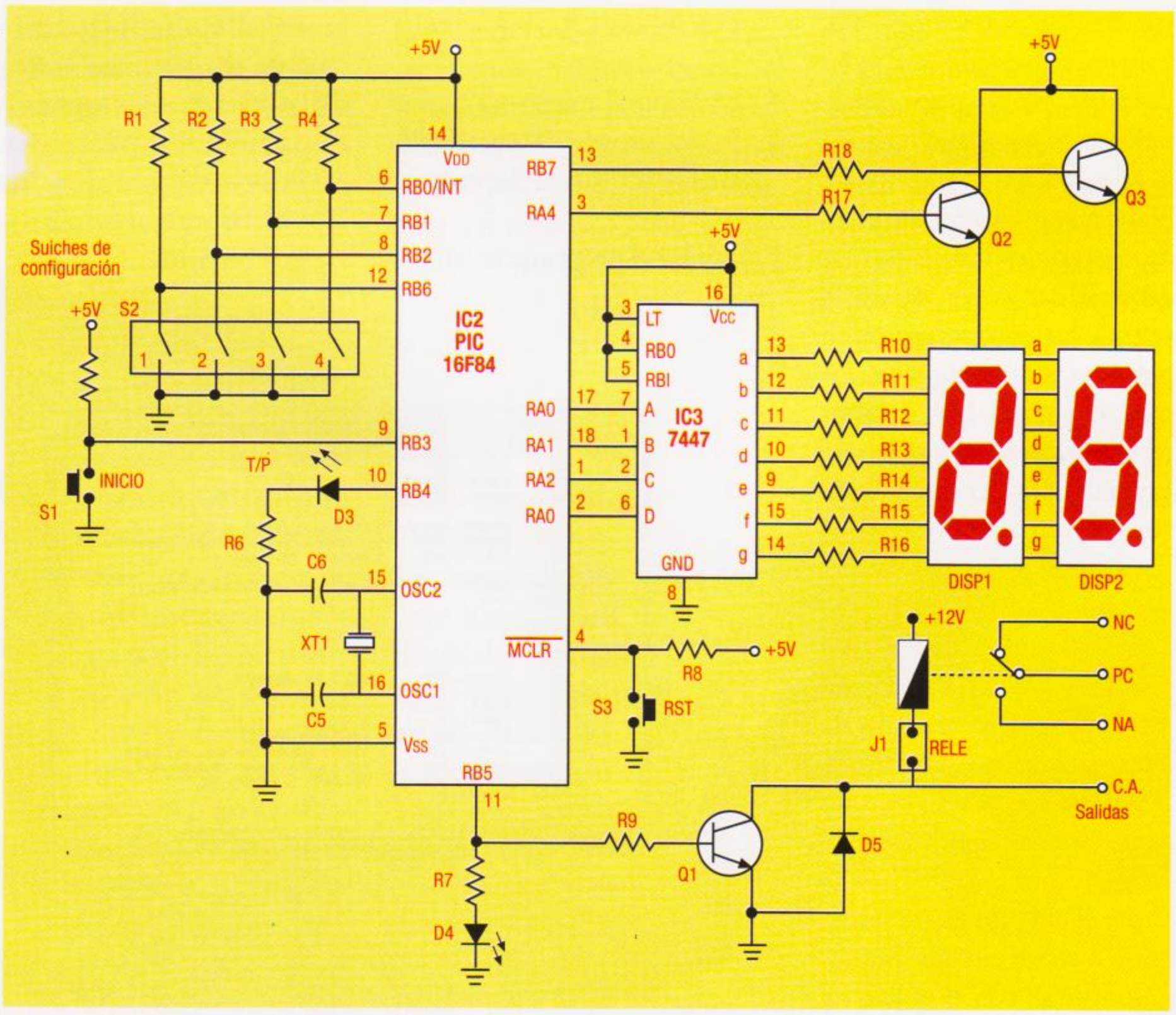


Figura 1.7 *Diagrama esquemático de un proyecto con microcontrolador*

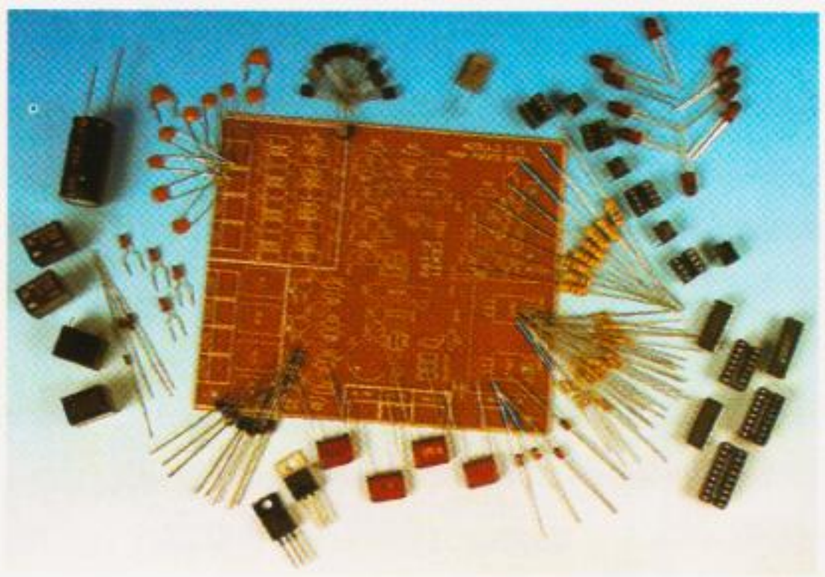


Figura 1.8 *Componentes para un proyecto con microcontrolador*

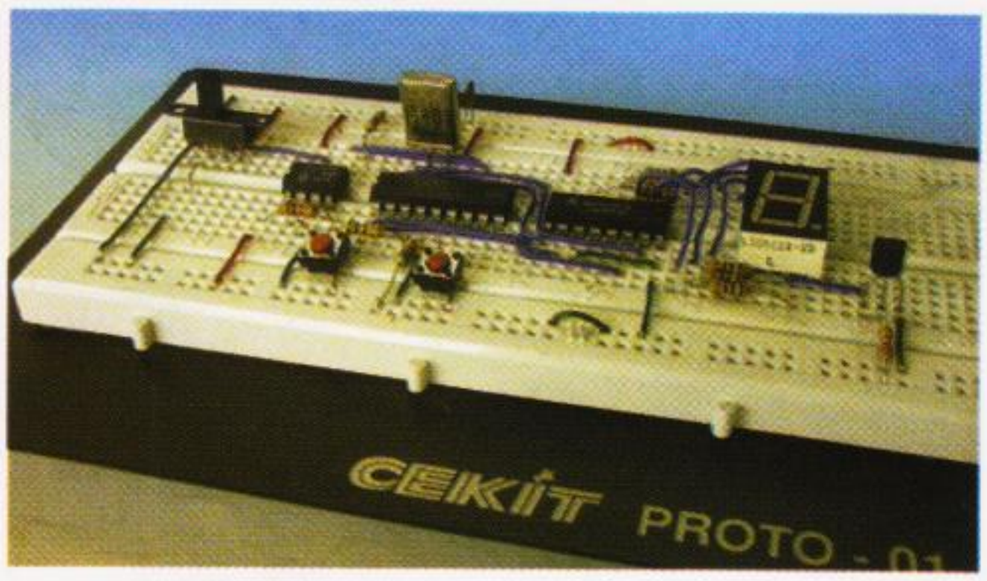


Figura 1.9 *Montaje del proyecto en un tablero de conexiones (protoboard)*

4) Diseñar el diagrama esquemático, **figura** 1.7, el cual debe incluir todos los componentes externos al microcontrolador; además, se deben considerar las características eléctricas de éste (capacidades de corriente por puerto, por pin, voltajes, velocidades, etc.) y las de las cargas o circuitos conectados a los puertos.

5) Elaborar una lista completa de materiales con todas las especificaciones de cada uno de los componentes, lo más detallada posible. Lo más recomendable es elaborarla en Excel con el fin de asignarle los precios a cada componente y obtener el costo total del proyecto.

6) Con base en la lista anterior, conseguir todos los componentes ya sea en los suministros locales, nacionales y aún del exterior, si se trata de un componente especial, **figura** 1.8.

7) Armar el circuito en un *protoboard* o en un circuito impreso universal, con el objeto de tener una versión inicial o prototipo para comprobar el diseño y realizar los ajustes necesarios si no se cumplen los objetivos iniciales. El microcontrolador se puede instalar sin tener grabado el programa para dejar listas sus conexiones, **figura** 1.9.

8) Elaborar el diagrama de flujo del programa para facilitar la escritura del mismo.

9) Escribir el programa en un editor de texto para computadora, siempre pensando en el diagrama esquemático y en la forma en que están conectados los pines de los puertos, estableciendo si las cargas se activan con "unos" o "ceros", **figura** 1.10.

10) Utilizar un programa ensamblador para corregir errores de sintaxis y "traducir" las instrucciones a códigos binarios.

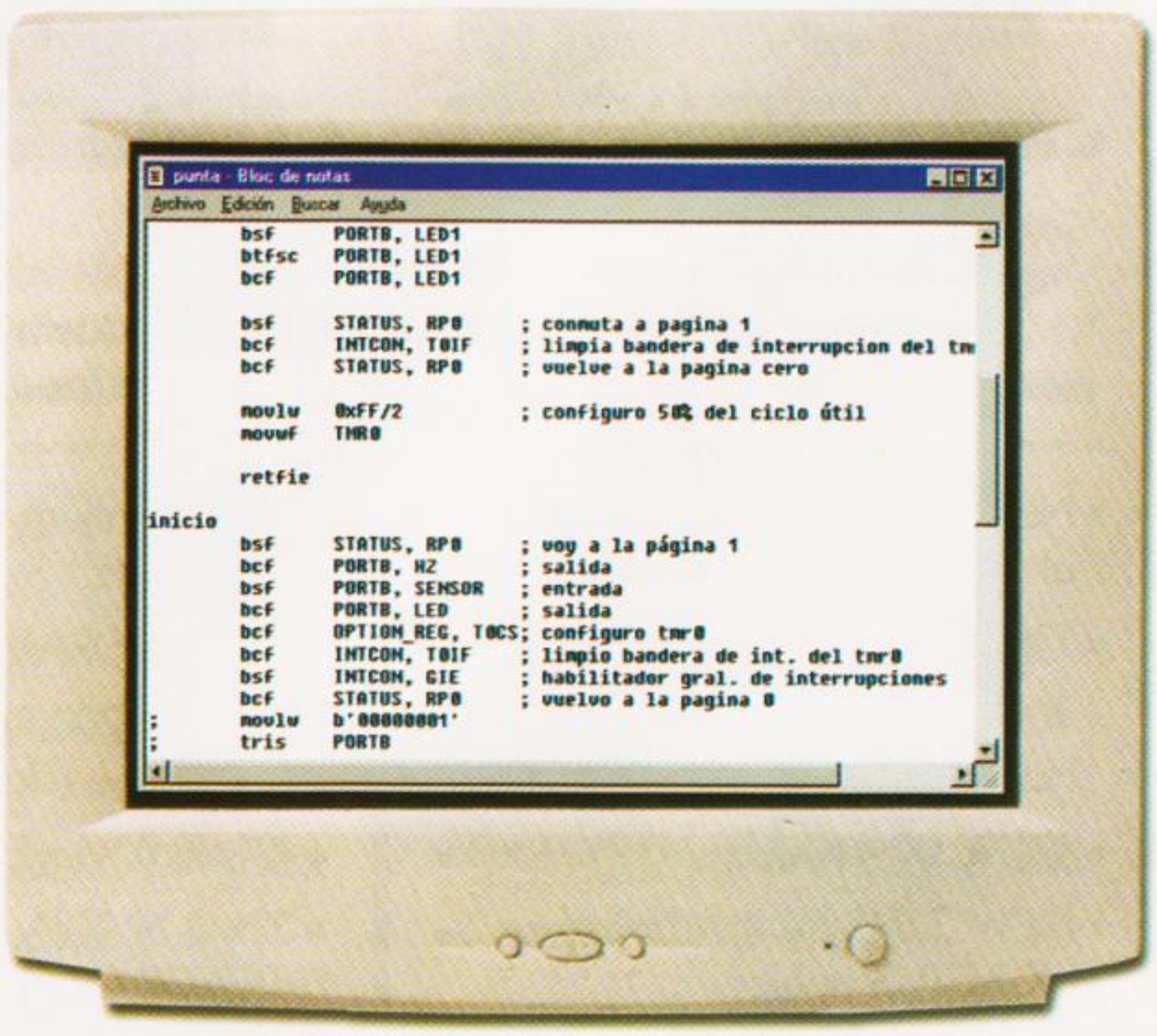


Figura 1.10 *Escribiendo el programa en un editor de textos*

Figura 1.11 *Programadores para microcontroladores*

11) Trasladar, por medio de un programador apropiado según la marca y referencia del microcontrolador, el código binario (.hex) obtenido desde la computadora a la memoria de programa del microcontrolador. A este proceso se le llama en algunas regiones, "quemar el microcontrolador", **figura 1.11.**

12) Montar nuevamente el microcontrolador en el *protoboard*, para verificar el funcionamiento del proyecto. En este momento se hacen todas las pruebas y se puede determinar si es necesario modificar el programa principal o algunas rutinas (por ejemplo, las de tiempo), para lo cual se regresa al editor de texto y se hacen los cambios pertinentes, repitiéndose el proceso hasta lograr al comportamiento deseado.

Es conveniente, por ejemplo, acudir a la ayuda de un programa simulador, con el cual se puede conocer el comportamiento de todos los registros de RAM y ROM cuando se desarrollan rutinas o cálculos matemáticos. Este paso se puede denominar la simulación fuera de línea (*off-line*). Luego se puede pensar en la ejecución de una simulación en línea (*on-line*), si se tienen las herramientas adecuadas o si se utiliza un microcontrolador que permita llevar a cabo esta tarea, **figura 1.12.**

Ella consiste en simular dentro del circuito y en los pines del microcontrolador, los programas o cambios que el usuario ejecute desde su computadora; aquí el microcontrolador aún no se encuentra programado. La última etapa es una emulación en tiempo real del programa, consistente en que la computadora, con ayuda de unos equipos especiales (emuladores), puede ejecutar el programa e imitar el comportamiento de los terminales según lo indique el programa que se va a grabar en el microcontrolador. Esto se hace a una velocidad tan alta, que si se compara a simple vista el funcionamiento del microcontrolador programado y la emulación, no se notarán diferencias.

Con estos procedimientos, se podría deducir que el proyecto funcionaría mejor con otros componentes o con otra configuración del circuito, lo cual hará que se rediseñe el proyecto, modificando buena parte de lo hecho hasta este punto.

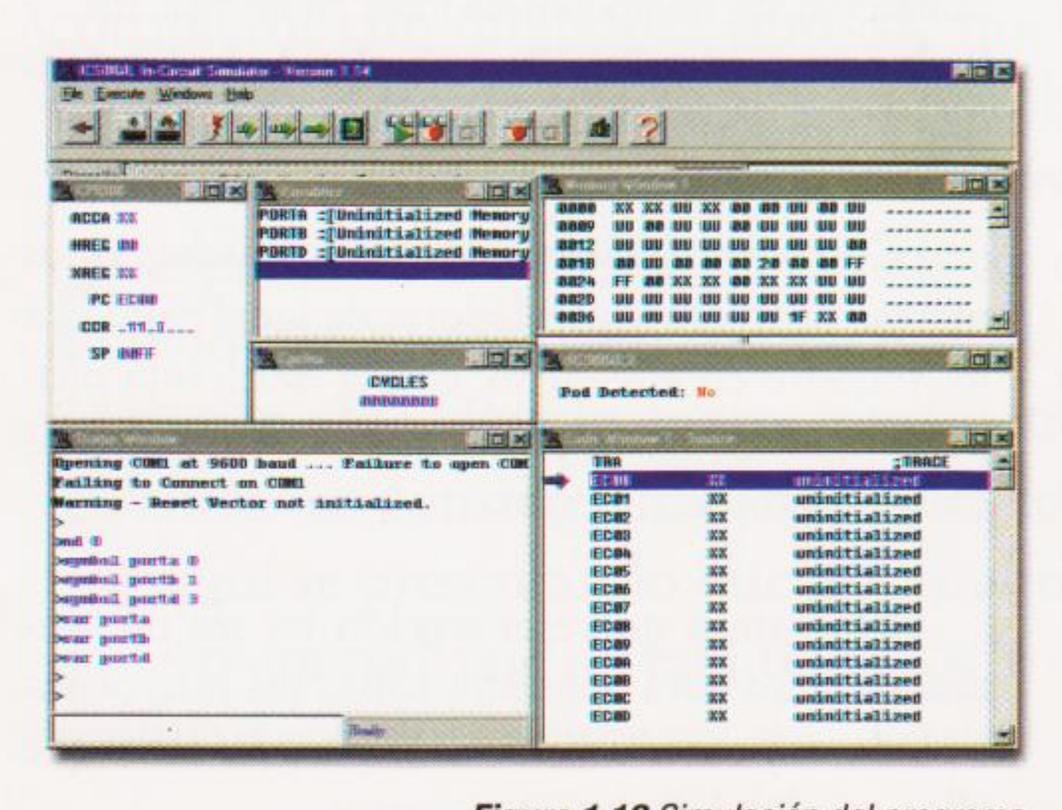


Figura 1.12 *Simulación del programa*

Figura 1.13 *Proyecto terminado con su circuito impreso y caja de montaje*

Superadas todas las etapas anteriores y funcionando de manera óptima, tanto el circuito como el programa, se pueden diseñar y construir el circuito impreso final y la caja de montaje, que alojará los componentes de manera definitiva dando por concluído el proceso, **figura 1.13.**

Para realizar los experimentos y construir los proyectos de este curso, es conveniente poseer conocimientos previos en electrónica digital, electrónica análoga y bases sobre el manejo de computadoras. Si se tienen deficiencias en alguna de estas áreas, les recomendamos acudir a los cursos que sobre estas materias ha publicado CEKIT S.A.

Selección del tipo de microcontrolador

Cuando se desea construir un aparato utilizando un microcontrolador como elemento central, se deben tener en cuenta varios aspectos. Para empezar, se deben conocer los alcances y requerimientos del proyecto con el fin de escoger el microcontrolador más adecuado.

Esta decisión, se puede tomar basándose en la lista de requisitos que se tienen para la aplicación. Si compara esta lista con las hojas de descripción de los distintos microcontroladores, seguramente podrá identificar fácilmente el dispositivo que más se acomode a su proyecto.

Entre las características que se deben tener en cuenta, se tienen:

- Entradas/salidas paralelas programables
- Canales análogos programables
- Salidas de PWM para control de motores
- Condiciones para la generación de interrupciones
- Cantidad y tamaño de los temporizadores
- Cantidad de memoria
- Tipo de memoria de programa (ROM, EPROM, PROM y EEPROM)
- Cómputo del punto flotante
- Niveles de voltaje y corriente que pueden manejar

En la sección de teoría se explicará detalladamente, a través de las diferentes lecciones, de que se trata cada uno de estos términos con el fin de que el alumno los pueda analizar correctamente; así que no se preocupe si no los entiende bien por ahora. Al avanzar en el curso, los irá conociendo y entendiendo mejor.

Para facilitar esta tarea, cada fabricante de microcontroladores tiene disponibles una serie de manuales de datos técnicos donde se describen las principales características de cada uno de sus dispositivos.

Así mismo, cada empresa tiene su página *web* en la internet con una información inmediata y muy completa sobre todos los microcontroladores disponibles para el diseño de proyectos de todo tipo. Dentro de esta información se encuentran las "Notas de aplicación" (*Application Notes*), estas son proyectos completos, incluyendo el diagrama y el programa, que brindan soluciones a muchas de nuestras necesidades.

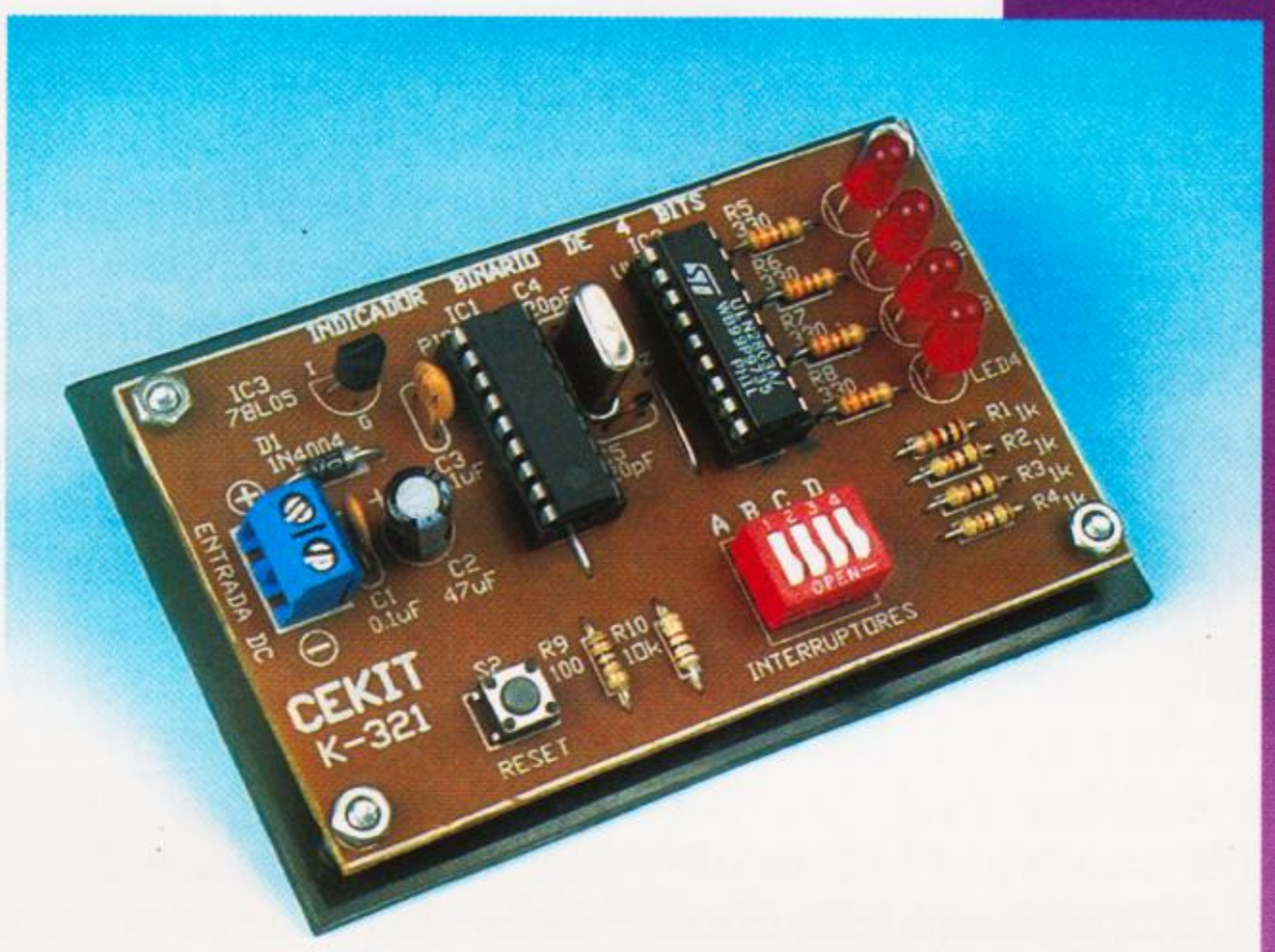
En la práctica, seguramente se encontrarán, comparando cada casa fabricante, características muy similares en cada una de las líneas de productos. La decisión de cual se va a utilizar debe entonces contemplar aspectos como el acceso a las herramientas de desarrollo, el precio, el soporte del fabricante, la disponibilidad cercana y la familiaridad, y los conocimientos que tenga el usuario frente a una marca determinada.

Esta sección de proyectos se ha enfocado en la utilización de microcontoladores Motorola, PIC y Basic Stamp, por estar ellos entre los de mayor difusión y comercialización a nivel regional y mundial.

Como en el mercado existen muchas otras marcas de microcontroladores de igual calidad a los tratados en este curso, con la estructura proporcionada, el lector podrá migrar fácil y rápidamente a otra tecnología.

Esperamos que esta sección les sea de mucha utilidad para aprender sobre los microcontroladores y tener una idea de las tantas cosas que ellos pueden hacer para nosotros.

PROYECTO 1



LECTOR BINARIO DE 4 BITS

Por su sencillez, este es un proyecto muy apropiado para iniciarnos en la práctica de aplicaciones con microcontroladores. En este sistema se utilizan los puertos o líneas de entrada/salida de un microcontrolador, para lectura y salida de datos digitales.

Características

- Lectura de cuatro señales digitales de entrada provenientes de un interruptor *dipswitch*
- Salida de datos digitales a través de diodos LED
- Manejo de *buffer* para amplificación de la corriente
- Incluye fuente regulada

Funcionamiento

Para introducirnos de manera rápida en el manejo de los microcontroladores, y en este caso de los PIC, vamos a hacer un montaje en donde se conectan cuatro interruptores (tipo *dipswitch*) como entradas del microcontrolador y cuatro diodos LED como salidas. Este proyecto tiene la configuración típica de un sistema de control con microcontroladores: elementos de entrada, unidad de proceso y elementos de salida; tal como se puede apreciar en el diagrama de bloques de la **figura 1.1**. El programa se encarga de verificar el estado de los interruptores en cuanto a su nivel lógico (1 ó 0) ó de voltaje, y de acuerdo a esta lectura, se deben encender o no los diodos LED respectivos.

Este proyecto es fundamental para ejercitarnos en el manejo de los puertos y podemos en un momento dado reemplazar los *dipswitch* por diferentes sensores de entrada y los LED de salida, por otro tipo de actuadores como relés, triacs, electroválvulas, etc. La **figura 1.2** muestra el diagrama esquemático del circuito.

Figura 1.1 *Diagrama de bloques del proyecto*

Nótese que los interruptores tienen resistencias de 1 KΩ conectadas al positivo de la fuente de alimentación; estas sirven para fijar un nivel alto cuando el interruptor correspondiente no esté haciendo contacto. En este caso, cuando no se encuentra cerrado ningún interruptor, el microcontrolador lee, en sus líneas de entrada, "unos" y cuando alguno se encuentre cerrado leerá un "cero", **figura 1.3**. Por otra parte, para encender los diodos LED se utiliza como *buffer* o acoplador un circuito integrado ULN2803, el cual tiene un conjunto de transistores que invierten el nivel de voltaje (si es alto lo pasan a

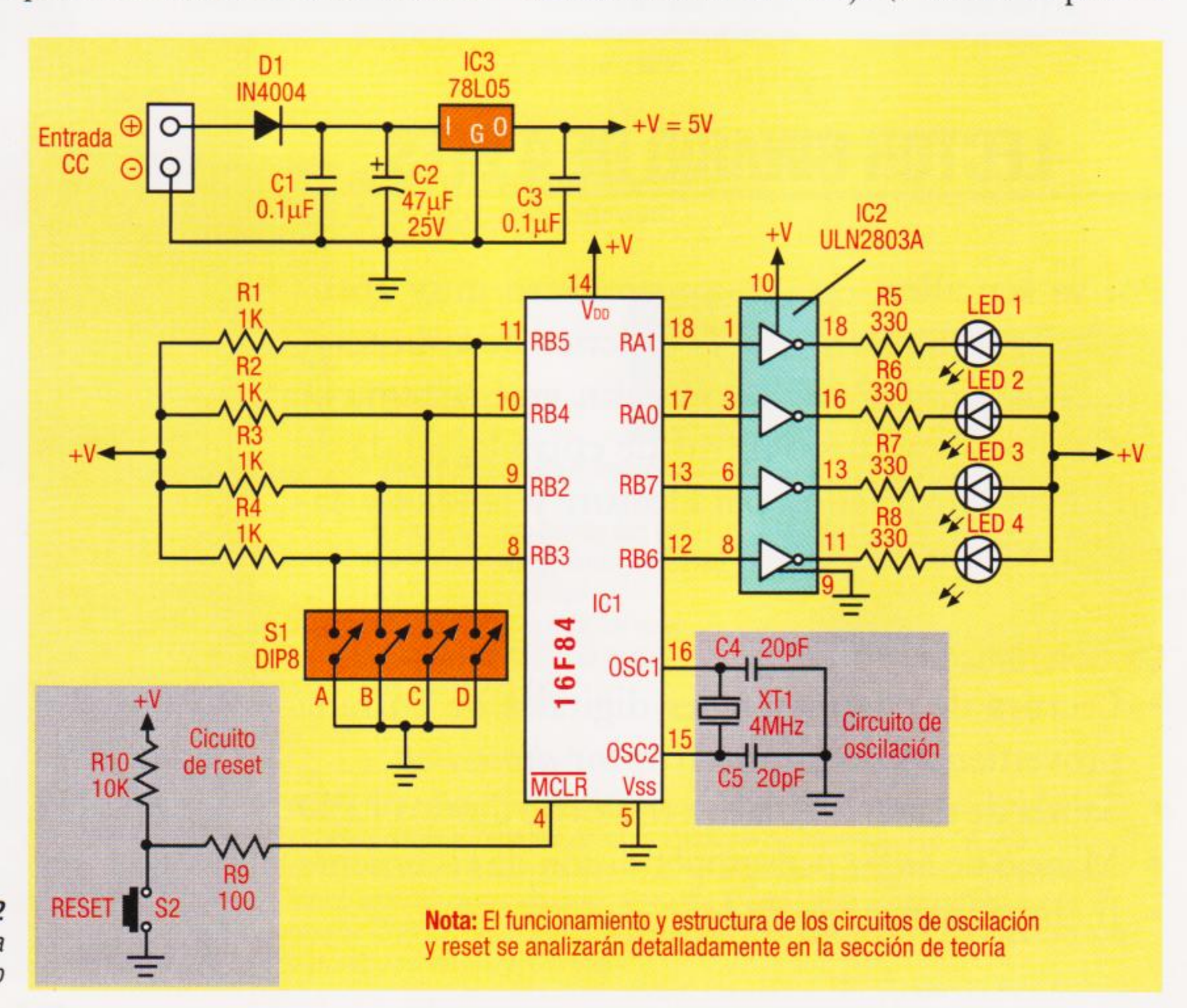


Figura 1.2 *Diagrama esquemático*

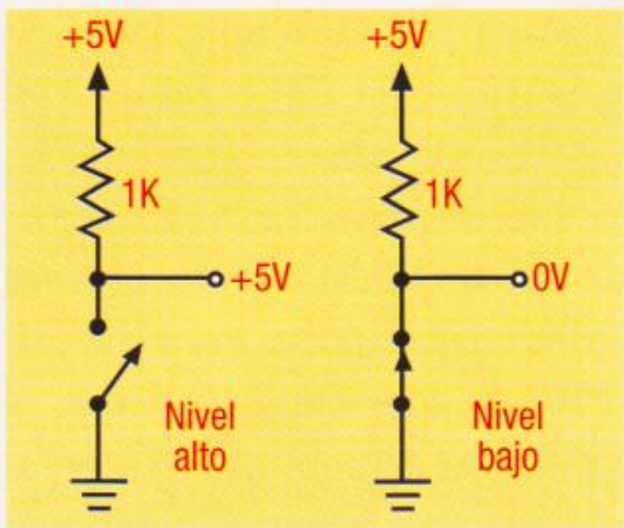


Figura 1.3 *Obtención de los niveles lógicos en las líneas de entrada*

bajo y viceversa) y amplifican la corriente. Por lo tanto, el nivel de salida para encender un LED debe ser positivo.

La fuente regulada de voltaje

Aunque los microcontroladores PIC pueden ser alimentados con voltajes entre 3 y 6V, es recomendable suministrarles un voltaje estable y de bajo rizado, dentro de dicho rango, para que el comportamiento de los circuitos que poseen en su interior sea el esperado y no ocasione algún problema en la ejecución de los programas.

El diseño de la fuente incluye al regulador positivo de 5V, de la serie de reguladores integrados de tres terminales 78L05, cuya capacidad de suministro de corriente es del orden de 100mA, suficiente para alimentar el circuito.

Para mejorar la estabilidad y garantizar el buen desempeño del regulador, éste debe estar acompañado de los condensadores C1, C2 y C3. El diodo rectificador de propósito general D1, que abre o cierra el paso de la corriente hacia el circuito dependiendo de la polaridad entre sus terminales (ánodo y cátodo), se ha incluido en la entrada como una protección para el circuito de este circuito si hay una inversión prolongada en la polaridad del voltaje de alimentación.

Los diodos emisores de luz LED (*Light Emitting Diodes*)

Son un tipo de diodos que se usan generalmente como indicadores luminosos y como visualizadores, **figura 1.4**. Para que un diodo LED pueda emitir luz, es necesario que sea polarizado directamente con un voltaje positivo entre ánodo y cátodo superior a 1.5V aproximadamente.

Las principales ventajas de los LED como emisores de luz es su tecnología de fabricación en estado sólido, lo cual implica que su desgaste o vida útil es mucho más amplia que la de los indicadores luminosos incandescentes. Bajo condiciones de polaridad directa, el diodo LED brilla con una intensidad luminosa proporcional a la magnitud de la corriente que pasa a través de él, la cual debe ser limitada a menos de 50mA (típicamente 5mA) para evitar su destrucción por calentamiento.

El microcontrolador

Antes de continuar con la explicación del funcionamiento de este proyecto, analicemos brevemente las características del microcontrolador utilizado. Se trata del muy conocido PIC16F84 de Microchip, cuya distribución de pines la vemos en la **figura 1.5**. Sus principales características son:

- Memoria de programa Flash de 1K x 14
- Memoria de datos RAM de 68 bytes
- Memoria de datos EEPROM de 64 bytes
- PILA de 8 niveles
- Cuatro tipos de fuentes de interrupción
- 35 instrucciones
- Encapsulado de 18 pines
- Frecuencia de trabajo máxima de 10 MHz
- Un solo temporizador
- 13 líneas en total de entrada/salida (I/O)
- Corriente máxima absorbida: 80 mA en el puerto A y 150 mA en el puerto B
- Corriente máxima suministrada: 50 mA en el puerto A y 100 mA en el puerto B
- Corriente máxima absorbida por línea: 25 mA
- Corriente máxima suministrada por línea: 20 mA

Memoria de programa

Es una memoria de 1Kbyte de tamaño con palabras de 14 bits.

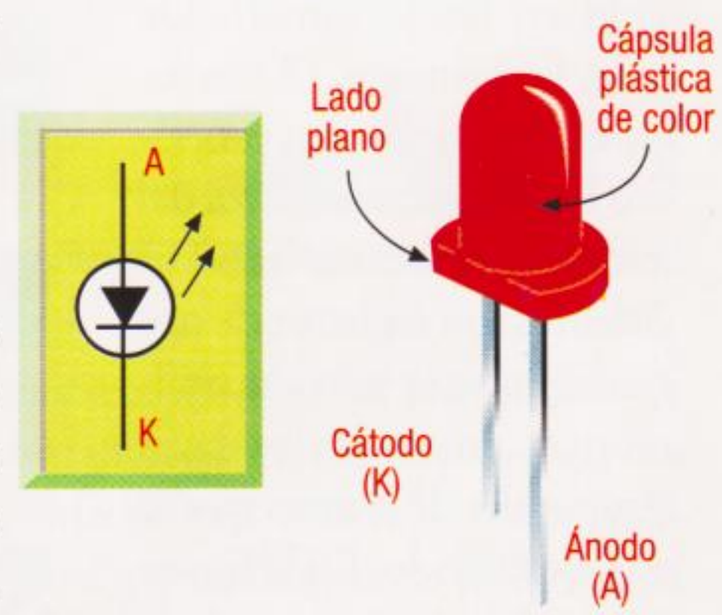


Figura 1.4 *Los diodos LED (aspecto físico y símbolo)*

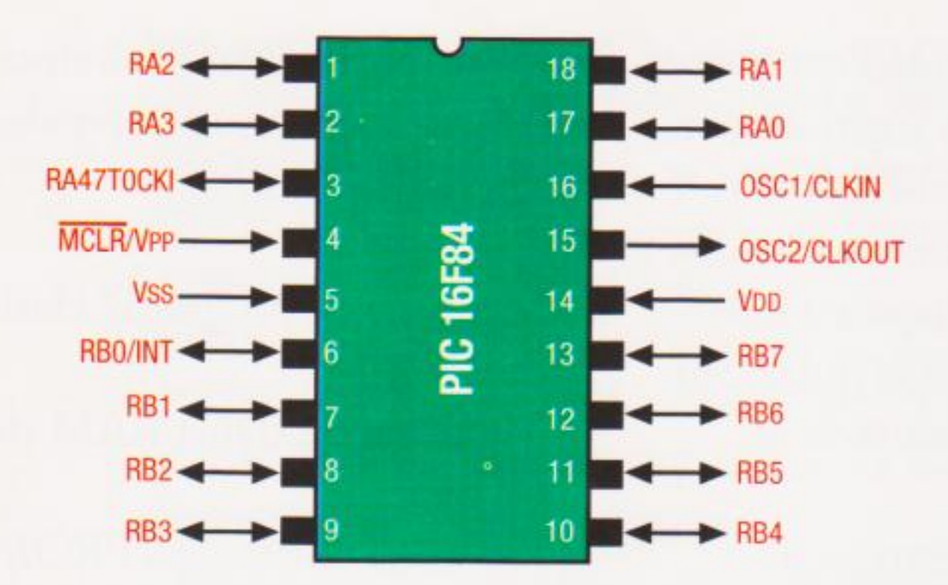


Figura 1.5 *Distribución de pines del microcontrolador 16F84*

Como es del tipo FLASH se puede programar y borrar eléctricamente, lo que facilita el desarrollo de los programas. En la **figura 1.6** vemos el mapa o distribución de esta memoria; en ella se puede apreciar el espacio de la pila y las ubicaciones específicas que deben ocupar cada una de las porciones del programa como el *reset* (0000H), el vector de interrupción (0004H) y la dirección de inicio del programa (0005H).

En la numeración 0000H, 0004H y 0005H, las primeras cuatro cifras indican una posición de memoria (0000, 0004 y 0005) y la letra H al final, indica que están escritas en el sistema de numeración hexadecimal, que es muy utilizada para este tipo de datos.

Memoria RAM

El PIC16F84 puede direccionar hasta128 posiciones de memoria RAM, pero sólo tiene establecidas físicamente las primeras 80 (00H-4FH en hexadecimal). De éstas, las primeras 12 son registros que tienen un propósito especial en el control del microcontrolador y las 68 siguientes son registros de uso general que se pueden utilizar para almacenar los datos temporales de la tarea que se está ejecutando. En la **figura 1.7** se aprecia el mapa de la memoria RAM.

Dentro de este mapa podemos apreciar el registro de estado en la dirección 03H, **figura 1.8**. Este registro contiene el estado aritmético de la ALU, la causa del *reset* y los bits de preselección de página para la memoria de datos (RAM). Estos últimos son especialmente útiles cuando trabajamos con microcontroladores PIC de más capacidad que el 16F84. En esos dispositivos, la memoria de programa (ROM) se divide en varias "páginas" o bloques de programa y la memoria de datos se divide en varios bloques denominados "bancos".

Podemos observar que la memoria RAM, en el caso del PIC 16F84, se divide sólo en dos "bancos". Para tener acceso a cada uno de los registros, debemos decirle al programa que direccione el banco correspondiente según el registro que deseemos trabajar. Para cambiar de banco, se utiliza el bit cinco del registro de estados, poniéndolo en uno o en cero según corresponda.

Descripción del programa

Cuando se lee el estado de los *dipswitch* se debe invertir el valor leído, para asegurarse que el interruptor que está cerrado se convierta en una señal positiva para encender el LED correspondiente. En la **figura 1.9** se muestra el diagrama

Figura 1.6 *Mapa de la memoria de programa*

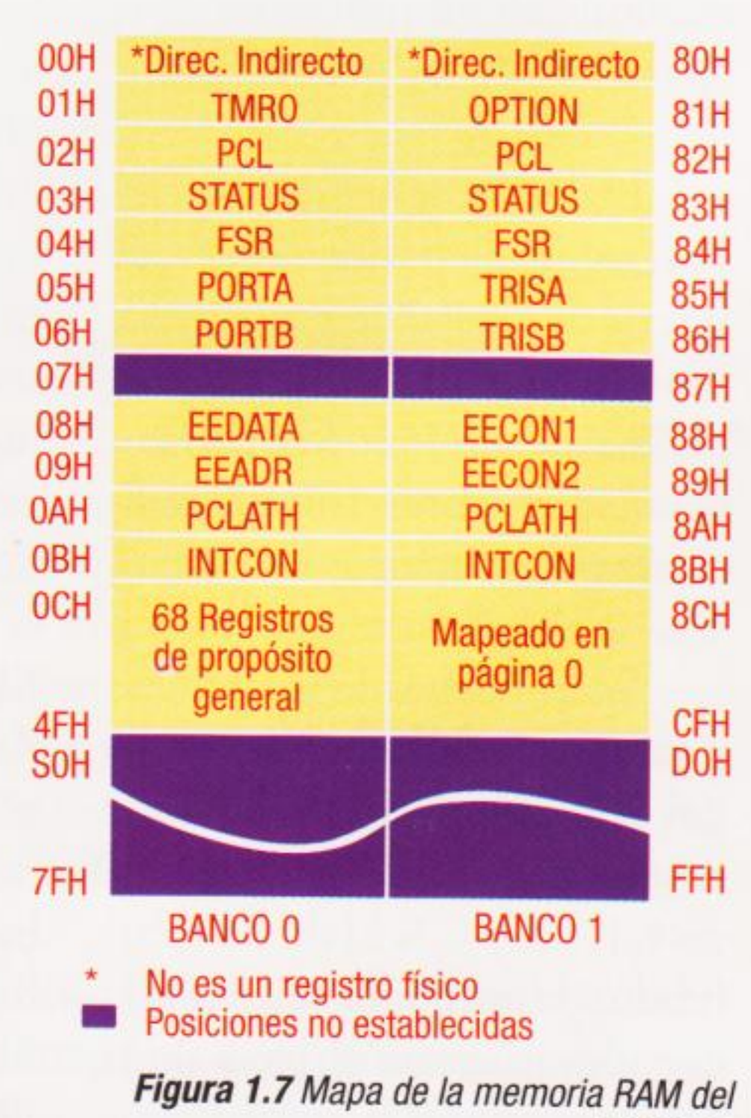


Figura 1.7 *Mapa de la memoria RAM del PIC16F84*

Registro: STATUS

IRP	RP1	RP0	TO	PD	Z	DC	C

bit 7 bit 0

Dirección: 03h
Condición de *reset*: 000??XXX

IRP: Selector de página para direccionamiento indirecto. Este *bit* no se utiliza efectivamente en el PIC16F84, por lo que se puede utilizar como un bit de propósito general.
RP1,0: Selectores de página para direccionamiento directo. Solamente RP0 se utiliza en el PIC16F84. RP1 se puede utilizar como un bit de propósito general.
TO: *Time Out* o *Bit* de finalización del temporizador. Se coloca en 0 cuando el circuito de vigilancia *Watchdog* finaliza la temporización.
PD: *Power Down* o *Bit* de bajo consumo. Se coloca en 0 por la instrucción *SLEEP*.
Z: *Zero o Bit* de cero. Se coloca en 1 cuando el resultado de una operación lógica o aritmética es cero.
DC: *Digit Carry o Bit* de acarreo de dígito. En operaciones aritméticas se activa cuando hay acarreo entre el bit 3 y el 4.
C: *Carry o Bit* de acarrreo. En instrucciones aritméticas se activa cuando se presenta acarreo desde el bit más significativo del resultado.

Figura 1.8 *Registro de estado*

de flujo correspondiente al ejercicio y en la **figura 1.10** el programa respectivo.

Cada uno de los bloques del diagrama de flujo se ha etiquetado con un número, al igual que cada una de las partes del código del programa se ha marcado con letras para ser referenciados dentro de la explicación que se dará a continuación; esta metodología se seguirá aplicando a lo largo de este curso.

Cuando escribimos un programa para un microcontrolador, debemos tener en cuenta el mapa de las direcciones de la memoria RAM que posea. Este mapa de direcciones debe contener los puertos, el registro de estado, el registro de interrupciones, etc., y debe estar incluido en todos los programas que desarrollemos; estas definiciones están dadas dentro de la sección A del programa a través de la directiva EQU. La función de esta directiva es definir una constante a un nombre en el programa; cualquier valor definido por EQU es fijo y no se puede modificar en ninguna parte del programa.

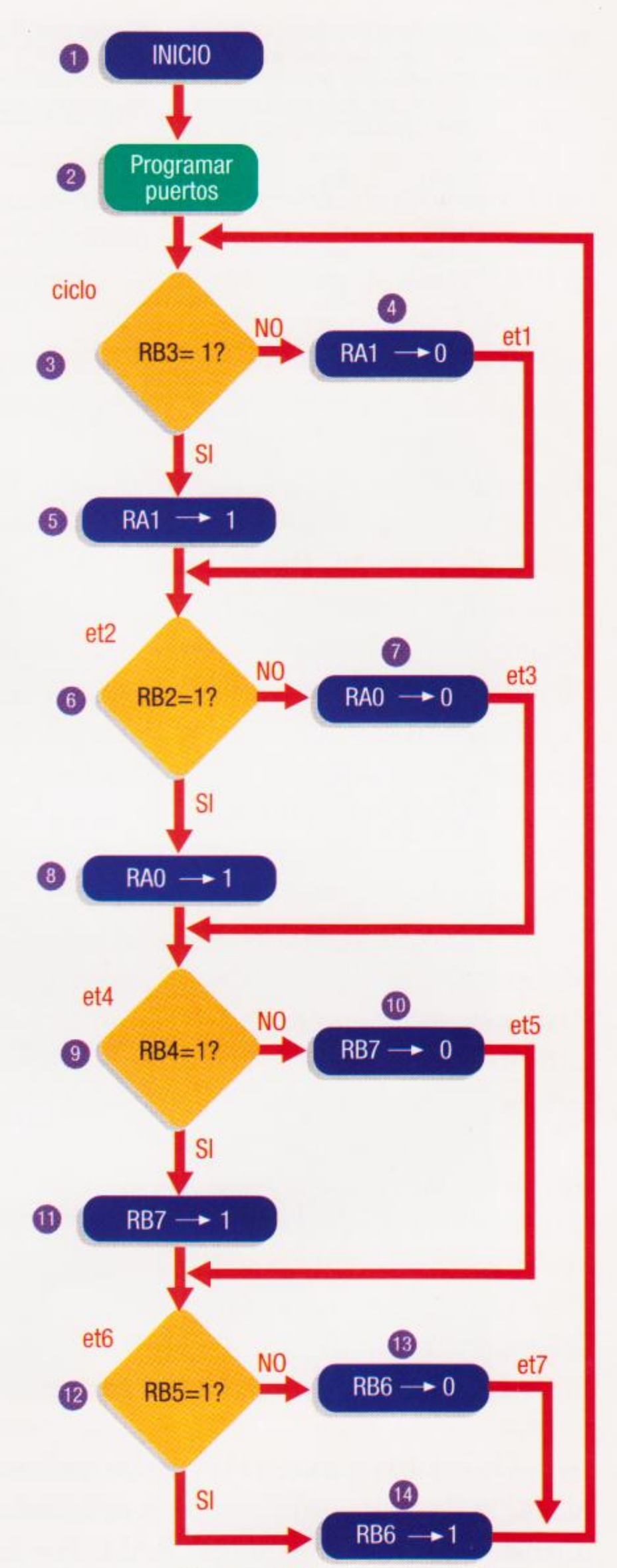


Figura 1.9 *Diagrama de flujo del programa*

Los nombres que se escogen son libres y a elección del programador; por ejemplo, *status* lo hubiéramos podido llamar "estado" o tener cualquier otro nombre, pero siempre se referirá a la dirección 03H de la memoria RAM que corresponde a ese registro (*STATUS*). Cada que utili-

```
;En caso de que un número se escriba D'15': significa número decimal
;En caso de que el número se escriba B'00010101': significa número binario
;En caso de que un número se escriba 15H: significa número hexadecimal
;Si no se especifica nada, se supone numeración hexadecimal
;definición de registros
pc          equ       02H
status      equ       03H
ptob        equ       06H        ;el puerto B está en la dirección 06 de la RAM
ptoa        equ       05H        ;el puerto B está en la dirección 06 de la RAM
trisb       equ       86H        ;registro de configuración del puerto B
trisa       equ       85H        ;registro de configuración del puerto A
w           equ       00H        ;indica que el resultado se guarda en W

reset       org       00         ;el vector de reset es la dirección 00
            goto      inicio     ;se salta al inicio del programa
            org       05         ;el programa empieza en la dirección de memoria 05

inicio      bsf       status,5   ;se ubica en el segundo banco de RAM
            movlw     3CH        ;se carga el registro W con 3C
            movwf     trisb      ;se programan los pines del puerto B
            movlw     03H
            movwf     trisa
            bcf       status,5   ;se ubica en el primer banco de memoria RAM

ciclo       btfss     ptob,3     ;chequeo el pin 3 del puerto B, (dip A)
            goto      et1        ;si está en cero salto a et1
            bsf       ptoa,1     ;si está en uno, pongo un uno en el pin RA1
et2         btfss     ptob,2     ;chequeo el pin 2 del puerto B (BIT B del dip)
            goto      et3        ;si está en cero, salto a et3
            bsf       ptoa,0     ;si está en uno, pongo un uno en el pin RA0
et4         btfss     ptob,4     ;chequeo el pin 4 del puerto B (BIT C del dip)
            goto      et5        ;si está en cero, salto a et5
            bsf       ptob,7     ;si está en uno, pongo un uno en el pin RB7
et6         btfss     ptob,5     ;chequeo el pin 5 del puerto B (BIT D del dip)
            goto      et7        ;si está en cero, salto a et7
            bsf       ptob,6     ;si está en uno, pongo un uno en el pin RB6
et8         goto      ciclo      ;repito todo el proceso

et1         bcf       ptoa,1     ;pongo cero en el pin 1 del puerto A
            goto      et2        ;regreso a chequear el punto B del dip
et3         bcf       ptoa,0     ;pongo cero en el pin 0 del puerto A
            goto      et4        ;regreso a chequear el punto C del dip
et5         bcf       ptob,7     ;pongo cero en el pin 7 del puerto B
            goto      et6        ;regreso a chequear el punto D del dip
et7         bcf       ptob,6     ;pongo cero en el pin 6 del puerto B
            goto      et8        ;regreso para continuar el ciclo
            end
;==================================================================
;           Fusibles de programación
;           Osc                    XT
;           Watchdog               OFF
;           Code protect           OFF
;           Power-Up-Timer         ON+
;           Micro.                 PIC16F84
;==================================================================
```

A

B

C

D

Figura 1.10 *Programa para el proyecto*

cemos la palabra *status* en el programa, el ensamblador entenderá que nos estamos refiriendo a la dirección 03H de la memoria RAM. Por facilidad de entendimiento, es muy conveniente asignar siempre nombres relacionados con la función de cada uno de los registros para que nos ayuden a recordarlos.

En el bloque B del programa, se observa una nueva directiva denominada ORG. Con ella establecemos la dirección, en la memoria de programa, del código que sigue a continuación y que se debe ubicar en la dirección que ella referencia. Si miramos el mapa de la figura 1.6, observaremos que esta parte del programa lo que hace es ubicar al código en su posición correcta, el vector de *reset* en la dirección 0000H y el inicio del programa en la dirección 0005H.

Cuando desarrollamos una aplicación con microcontroladores, uno de los primeros pasos que debemos seguir es configurar el funcionamiento

de los puertos, es decir si cada uno de los pines o terminales del microcontrolador va a servir para entrar datos o para originar salida de datos. En algunos casos, un mismo pin puede alternar su funcionamiento, pero esto lo analizaremos más adelante. En el bloque 2 del diagrama de flujo y en el bloque C del programa, se establece esta configuración. Esto se hace a través de los registros ***trisa*** y ***trisb***, los cuales son registros de ocho bits que configuran los puertos A y B; al colocar un "cero" en el bit correspondiente al pin, se configura como salida, mientras que si se escribe un "uno", el pin o terminal queda configurado como entrada, **figura 1.11**. En el programa observamos cómo se hace el cambio a la página uno para poder acceder a ***trisa*** y ***trisb*** para modificarlos.

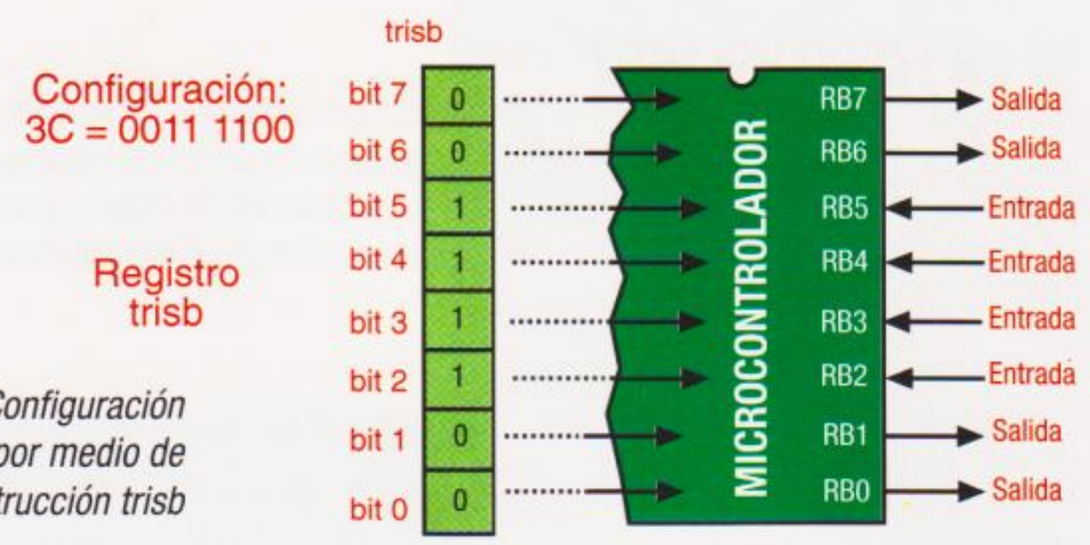


Figura 1.11 *Configuración del puerto B por medio de la instrucción trisb*

Los numerales 3 al 14 del diagrama de flujo y la parte D del programa, toman el valor leído de los interruptores chequeando uno a uno los bits RB3, RB2, RB4 y RB5. Según el estado de cada uno de esos pines, se debe sacar un "cero" o un "uno" por los pines RA1, RA0, RB7 y RB6, respectivamente. Para leer cada uno de los pines de entrada, se utiliza la instrucción ***btfss <registro>,b***, la cual lee el bit indicado por la letra **b** del registro indicado en **<registro>** (en este caso los puertos A y B), y se salta la siguiente instrucción si ese bit está en "uno"; si el bit está en "cero", el programa ejecuta la siguiente instrucción la cual corresponde a un ***goto<etiqueta>***. La instrucción ***goto*** permite saltar incondicionalmente varias posiciones hasta el lugar donde se encuentre la etiqueta que se referencia en **<etiqueta>**. Una vez se han leído los cuatro terminales del interruptor, y originado la salida correspondiente, el programa retorna al punto 2 del diagrama de flujo o etiqueta **ciclo** del programa, para repetir todo el proceso.

PARA SABER MÁS

Un ensamblador es una herramienta de desarrollo de *software* utilizada para trasladar un conjunto de instrucciones en ensamblador (llamado código fuente) a instrucciones que pueden ser entendidas por la computadora (denominado código de máquina). Las instrucciones en ensamblador son un conjunto de comandos establecidos a través de los cuales un programador humano puede ejecutar efectivamente instrucciones en lenguaje de máquina.

El lenguaje ensamblador de los microcontroladores PIC presenta tres componentes principales:
Instrucciones mnemónicas
Directivas
Macros

Mnemónicos son instrucciones que se trasladan directamente en código de máquina. Se usan para realizar operaciones matemáticas y lógicas sobre datos residentes en la memoria. También pueden mover datos entre registros, al igual que saltar a lo largo de la memoria de programa.

Las ***directivas*** son las que controlan la operación del ensamblador, pues orientan sobre cómo se deben tratar los mnemónicos, referenciar datos y en qué formato se originará el archivo de salida.

Un ***macro*** es una secuencia de comandos de instrucciones en ensamblador. Es un bloque de programa que realiza una función específica y por la tanto puede ser invocado cada vez que se necesite.

LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F84
1	IC2	Regulador fijo de +5V, LM78L05
1	IC3	Driver de transistores NPN, ULN2803
1	D1	Diodo rectificador de propósito general 1N4004
4	LED1, LED2, LED3 y LED4	Diodos LED de 5mm, rojos
2	C1 y C3	Condensadores cerámicos de 0.1uF - 50V
1	C2	Condensador electrolítico de 47uF - 25V
2	C4 y C5	Condensadores cerámicos de 20pF - 50V
1	XT1	Cristal de cuarzo de 4MHz
4	R1, R2, R3 y R4	Resistencias de 1KΩ, 1/4W
4	R5, R6, R7 y R8	Resistencias de 330 Ω, 1/4W
1	R9	Resistencia de 100 Ω, 1/4W
1	R10	Resistencia de 10K Ω, 1/4W
1	S1	Interruptures tipo DIPswitch de 4 posiciones
1	S2	Pulsador pequeño de 4 pines, para circuito impreso
1	K-321	Circuito impreso CEKIT
2		Bases para circuito integrado de 18 pines
1		Conector de tornillo de 2 pines

Ensamblaje del proyecto

Aunque este proyecto es simple y emplea pocos componentes, el proceso de ensamblaje se debe hacer de una forma técnica contando con las herramientas básicas tales como el cautín, las pinzas, el cortafrío, el pelacables y la soldadura de estaño; usando la guía de ensamblaje dibujada sobre la cara de los componentes en el circuito impreso, y siguiendo cuidadosamente estos pasos:

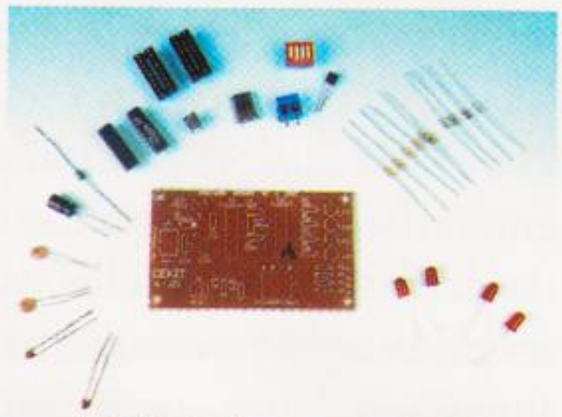


Figura 1.12 *En la figura se muestra el conjunto de componentes electrónicos, elementos electromecánicos y accesorios, que se requieren para ensamblar el lector binario de 4 bits.*

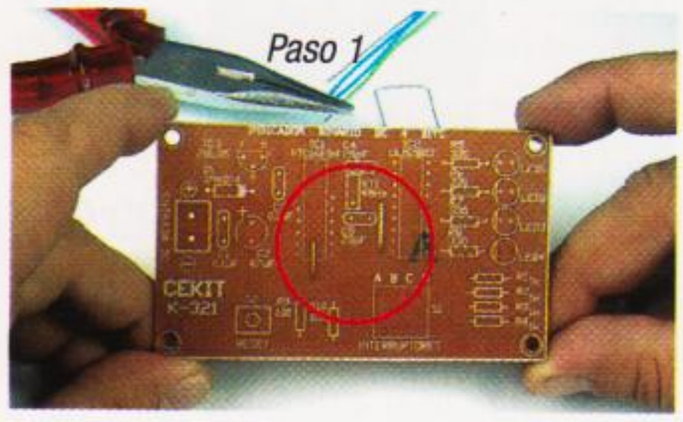


Figura 1.13 *Usando alambre telefónico desnudo número 24, construya con las pinzas los dos puentes de alambre y suéldelos al nivel de la superficie de la tarjeta.*

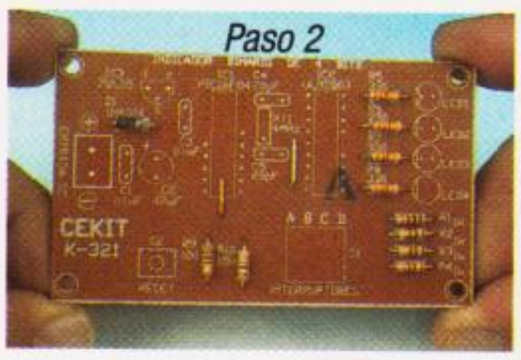


Figura 1.14 *Ubique y suelde todas las resistencias y el diodo rectificador.*

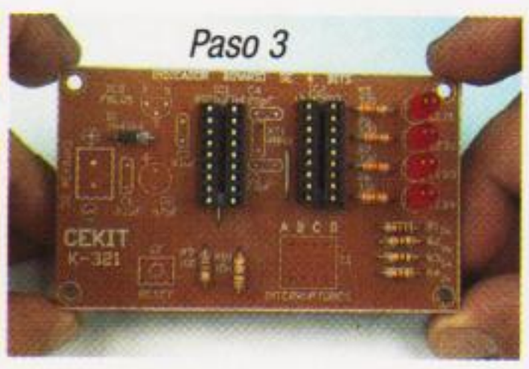


Figura 1.15 *Suelde a continuación las bases y los diodos LED, orientando cuidadosamente el ánodo y el cátodo.*

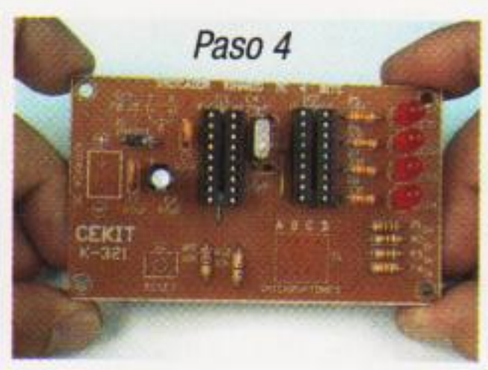


Figura 1.16 *Fije ahora, con soldadura, los condensadores cerámicos primero, luego el condensador electrolítico y el cristal de 4 MHz.*

Figura 1.17 *Suelde el regulador IC3, el pulsador S2, el interruptor tipo DIP y el conector de entrada.*

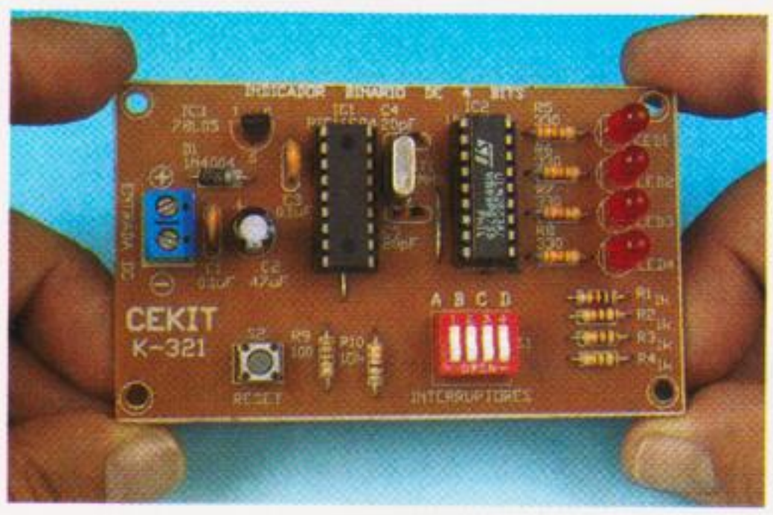


Figura 1.18 *En la fotografía, se muestra el aspecto del proyecto ensamblado.*

Figura 1.19 *Con una batería o fuente de 9V CC, alimente la tarjeta sin el microcontrolador, y mida con un multímetro el voltaje entre los pines 14 y 5 de la base del microcontrolador (IC1), el cual debe ser +5V CC, si el circuito está regulando bien.*

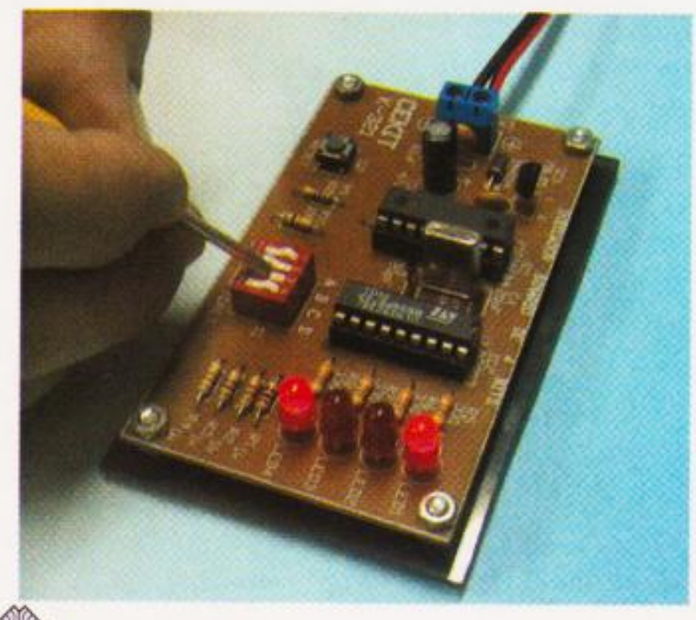
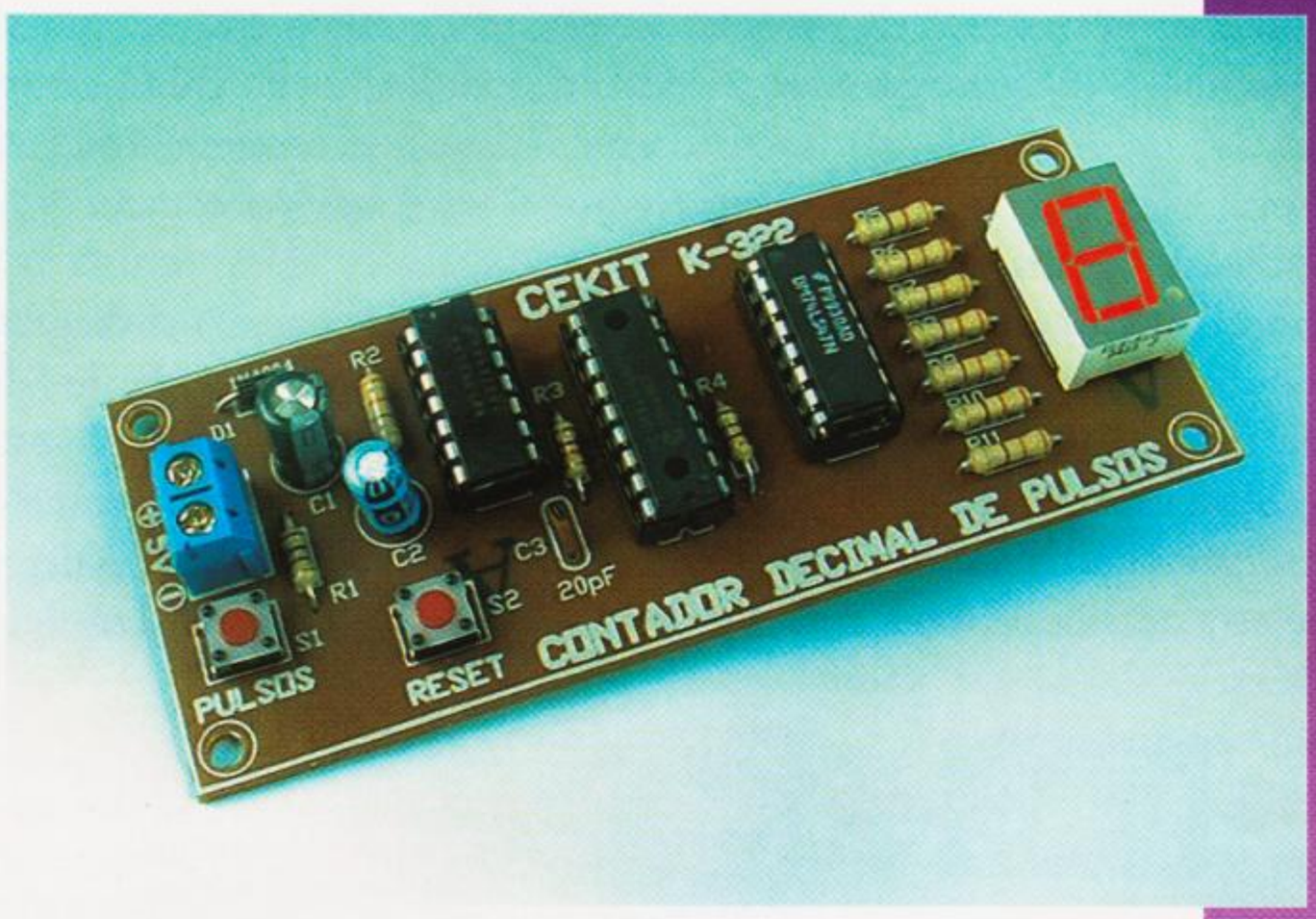


Figura 1.20 *Instale en su base el microcontrolador 16F84 programado; alimente el circuito y conmute cada uno de los interruptores del interruptor tipo DIP (A,B,C,D), y observe que cada uno de los diodos LED se enciendan o apaguen en respuesta a la acción sobre su correspondiente interruptor. Si esto ocurre, el circuito está trabajando bien. Si no es así, revise muy bien la posición de los componentes, todas las soldaduras y el programa del microcontrolador.*

PROYECTO 2



CONTADOR DECIMAL DE PULSOS

Por medio de este proyecto, tendremos un circuito que cuenta, de 0 a 9, los pulsos provenientes de un pulsador y los muestra en un *display* o indicador de siete segmentos. Aprenderemos a manejar rutinas de retardo y conoceremos el funcionamiento de un oscilador RC para un microcontrolador.

Características

- Lectura de una señal de entrada proveniente de un pulsador. Este elemento puede ser en un momento dado un sensor u otro sistema de detección.
- Cuenta los pulsos que se originan en la entrada.

Funcionamiento

En este segundo proyecto, aprenderemos conceptos y técnicas muy importantes para el desarrollo de aplicaciones con microcontroladores PIC. Este montaje tiene por objeto visualizar en un *display* de siete segmentos el conteo de las veces que se acciona un pulsador (S1). Como por ahora trabajaremos con un solo *display*, visualizaremos el conteo desde cero (0) hasta nueve (9) y luego iniciaremos de nuevo.

Muchas veces, cuando desarrollamos circuitos prácticos con microcontroladores y distintas clases de sensores, nos encontramos en la situación de tener que detectar señales digitales de corta duración. Para estos casos, existen algunos métodos que nos permiten leerlas correctamente, los cuales trataremos en esta ocasión al tener como entrada al sistema un pulsador. En la **figura 2.1** observamos el diagrama de bloques de este proyecto; la unidad de entrada la conforma el pulsador S1, la unidad de proceso es un microcontrolador PIC16F84, el cual hemos seleccionado por las ventajas que tenemos al trabajar con un dispositivo con memoria EEPROM y finalmente, la unidad de salida está compuesta por un circuito decodificador de BCD a siete segmentos y un *display* de siete segmentos, en el cual observaremos el resultado del conteo.

En la **figura 2.2** se muestra el diagrama esquemático del circuito. La entrada de S1 se ha conectado a una línea del puerto A, en este caso a RA0 (pin 17) y las señales para el *display* salen por los pines RB0, RB1, RB2 y RB3 del puerto B. En estos cuatro pines se origina el código BCD

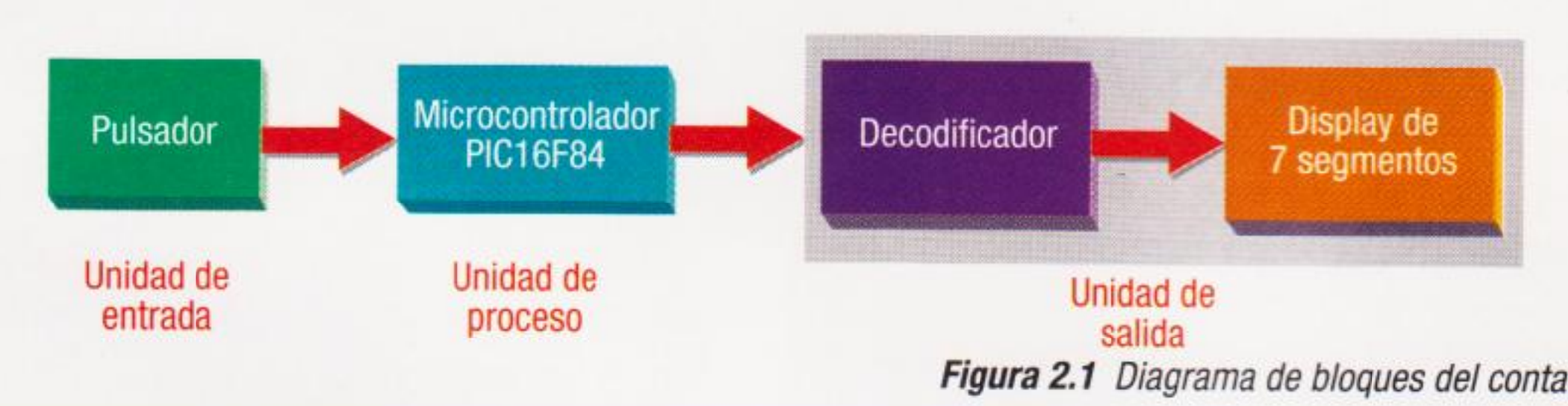


Figura 2.1 *Diagrama de bloques del contador de pulsos*

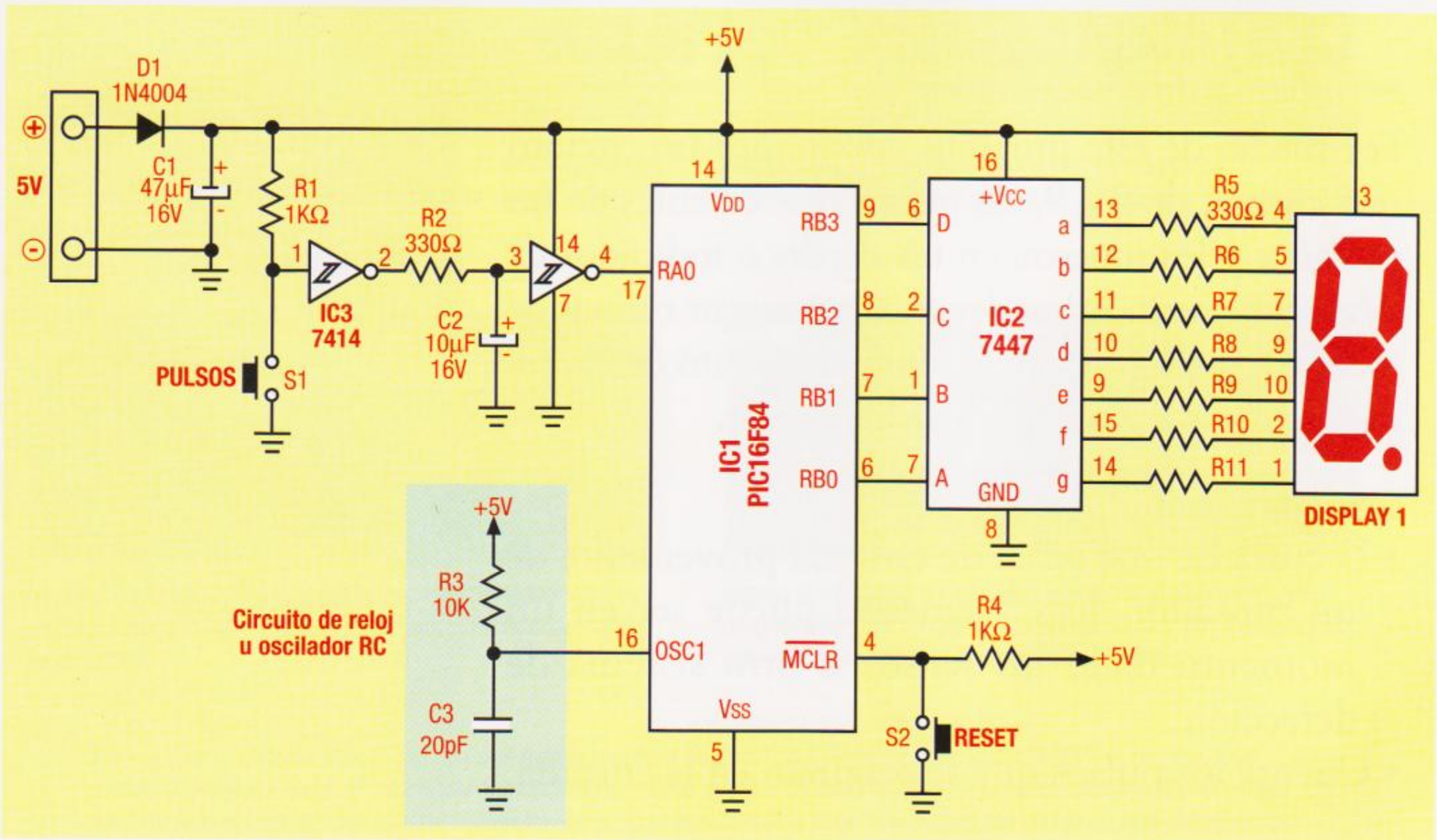


Figura 2.2 *Diagrama esquemático del circuito*

correspondiente al número; el circuito integrado TTL 7447 se encarga de traducir este código y entregar la representación en el formato de siete segmentos. Esta es una técnica sencilla que nos permite ahorrar líneas (pues sólo necesitamos cuatro líneas para manejar los siete segmentos) al trabajar con este tipo de visualizadores.

Notemos que el pulsador S1 tiene una resistencia de 1 KΩ conectada al positivo de la fuente de alimentación; por esta razón, cada vez que el pulsador S1 se oprime, se origina un estado lógico bajo (0 voltios) en la entrada de la primera compuerta (pin 1); de lo contrario, dicha entrada siempre permanecerá en estado lógico alto (5 voltios).

Circuito de reloj

Todo circuito con este tipo de microcontrolador, y en general con la gran mayoría de ellos, debe llevar un circuito externo de reloj que puede ser un cristal, un resonador o una red RC, tema que se explicará ampliamente en la sección de teoría. En este caso se ha utilizado como circuito oscilador una red RC, con una resistencia de 10 KΩ y un condensador de 20 pF. Esto se hace para que el sistema sea más económico y teniendo en cuenta que este tipo de oscilador se puede utilizar en aplicaciones donde no se requiere mucha velocidad o precisión de tiempos. ***Al incorporar esta red RC como circuito oscilador, se debe configurar esta elección en el momento de programar el microcontrolador, habilitando la opción que indica que este tipo de oscilador es el que se va a utilizar, figura 2.3.***

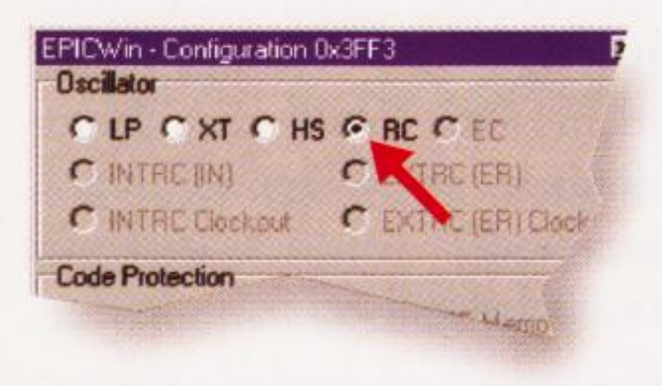


Figura 2.3 *Habilitación del oscilador RC al momento de programar el microcontrolador. Esta opción forma parte de los "fusibles" de programación que se deben configurar*

Esta opción forma parte de los "fusibles" de programación, los cuales son las opciones con las que se le dice al *software* programador que grabe el programa. En la **figura 2.4** se indican las frecuencias de oscilación que originan en el microcontrolador, distintos valores de R y C.

R	C	Frecuencia (Aprox.)
10K	20pF	62.5 KHz
10K	220pF	80 KHz
10K	0.1μF	85 Hz

Figura 2.4 *Frecuencias de oscilación originadas por distintos valores en la red RC*

El *display* de 7 segmentos

Es un dispositivo semiconductor que se utiliza para visualizar los números del 0 al 9, así como algunas letras y símbolos especiales. Los segmentos, cada uno de los cuales está formado por un LED con forma rectangular, están organizados en forma de "8" y se designan como se indica en la **figura 2.5**. La mayoría incluyen un LED adicional que sirve como punto decimal.

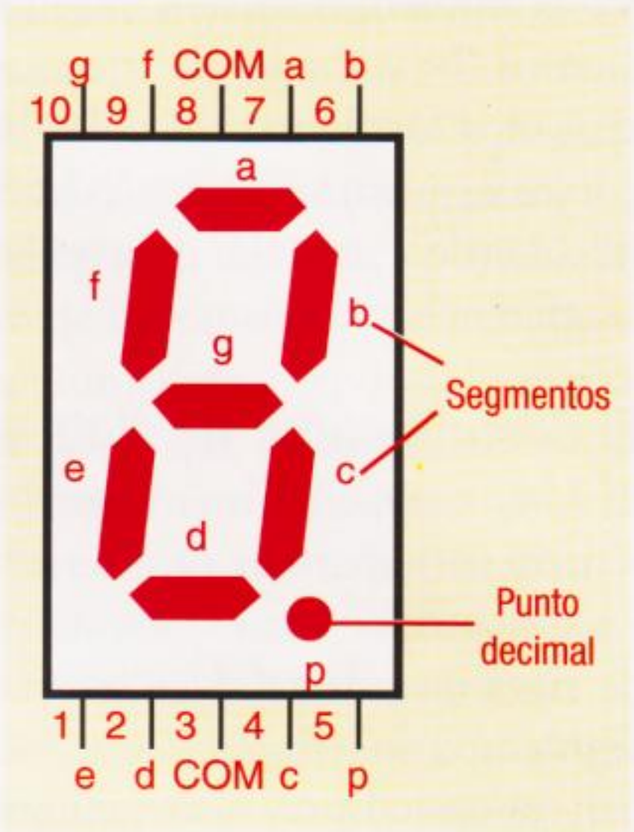


Figura 2.5 *Designación de los segmentos del display y numeración sus terminales*

Los *displays* pueden ser de ánodo o de cátodo común, dependiendo de como estén conectados internamente los segmentos. El utilizado en este proyecto es de ánodo común. En este caso, todos los ánodos de los LED están conectados entre sí, formando un ánodo común, y todos los cátodos están libres. En un *display* de cátodo común, sucede todo lo contrario.

Notemos que externamente disponemos de 10 terminales de conexión: dos para el ánodo común (**COM**), siete para los cátodos de los segmentos (**a, b, c, d, e, f, g**) y uno para el cátodo del punto decimal (**P**). Los terminales se numeran del 1 al 10 como se indica en la figura 2.5. Para visualizar un carácter, por ejem-

plo el número 4, deben encenderse ciertos segmentos (**b, c, f** y **g**, en este caso) y mantenerse apagados los demás. El color de la luz emitida por los segmentos del *display* puede ser rojo, amarillo, verde y naranja, dependiendo del diseño y la referencia del mismo.

El decodificador de siete segmentos

El 7447 es un circuito integrado digital TTL diseñado para manejar directamente un *display* de 7 segmentos de ánodo común; por ello, cada uno de sus pines de salida, destinados a operar los segmentos del *display,* es una fuente de corriente con capacidad para manejar hasta 10mA por segmento. Para manejar el *display*, el circuito integrado tiene las salidas a, b, c, d, e, f y g, que se deben conectar a través de resistencias limitadoras de corriente con cada uno de los segmentos.

Para formar cualquiera de los números decimales a través de los segmentos del *display*, el decodificador cuenta con cuatro entradas binarias (A, B, C y D) en las cuales se representa el número deseado por medio del código BCD. Por ejemplo, para visualizar el número 5, las entradas del decodificador deben ser: A=1, B=0, C=1 y D=0.

El circuito eliminador de rebotes

Los interruptores mecánicos generan unas señales no deseadas de conexión y desconexión que originan falsas lecturas cuando se conectan como entradas en los circuitos con microcontroladores. Estas señales se denominan "de rebote" ya que se originan cuando dos piezas metálicas se acercan o alejan, o rebotan entre sí.

En este proyecto, el circuito eliminador de rebotes está compuesto por dos compuertas tipo *schmitt trigger* contenidas en el circuito integrado 7414, y un filtro pasabajos formado por la red R2 y C2. El propósito del circuito es amortiguar las conmutaciones rápidas que se presentan al accionar el pulsador S1, entregando un pulso único y limpio cada vez que éste se acciona. De esta forma, evitamos que el microcontrolador pueda detectar por su entrada (RA0), una falsa lectura y así ocasionar un incremento desordenado de la cuenta de los pulsos efectivos lo que se podrá apreciar en el *display*.

La primera compuerta de IC3 se encarga de efectuar un primer paso en el filtrado, al limitar la transición de 1 a 0 ó viceversa, por medio de la histéresis característica de la tecnología *schmitt trigger.* Un segundo nivel de amortiguamiento se obtiene cuando la salida de dicha compuerta se aplica al filtro pasabajos formado por R2 y C2. Los valores de estos dos elementos definen el nivel de amortiguamiento deseado.

Este filtro retarda la llegada del nivel lógico 0 ó 1 a la entrada de la segunda compuerta *schmitt trigger* de IC3, que se encarga de complementar el nivel de entrada, logrando así obtener un nivel lógico directo en la salida del eliminador de rebotes (pin 4 de IC3) en función del nivel de la entrada de pulsos.

El diodo D1 se utiliza dentro del circuito como un elemento de protección. Su función es actuar como una válvula unidireccional que permite el paso de la corriente solamente cuando se polariza directamente y opera como un circuito abierto cuando su polaridad está invertida. De esta forma, se protege el circuito contra las inversiones que puedan presentarse en la polaridad del voltaje de alimentación, que podrían dañar permanentemente el microcontrolador y los otros circuitos integrados.

Descripción del programa

En la **figura 2.6** observamos el diagrama de flujo del programa y en la **figura 2.7** el código fuente del mismo. Recordemos que en este curso la numeración de los bloques y del código del programa, se hicieron para poder identificar cada una de las partes dentro de la explicación.

En el bloque A del código fuente, se ha escrito una directiva con la cual se establecen algunas configuraciones con las que el programa ensamblador crea-

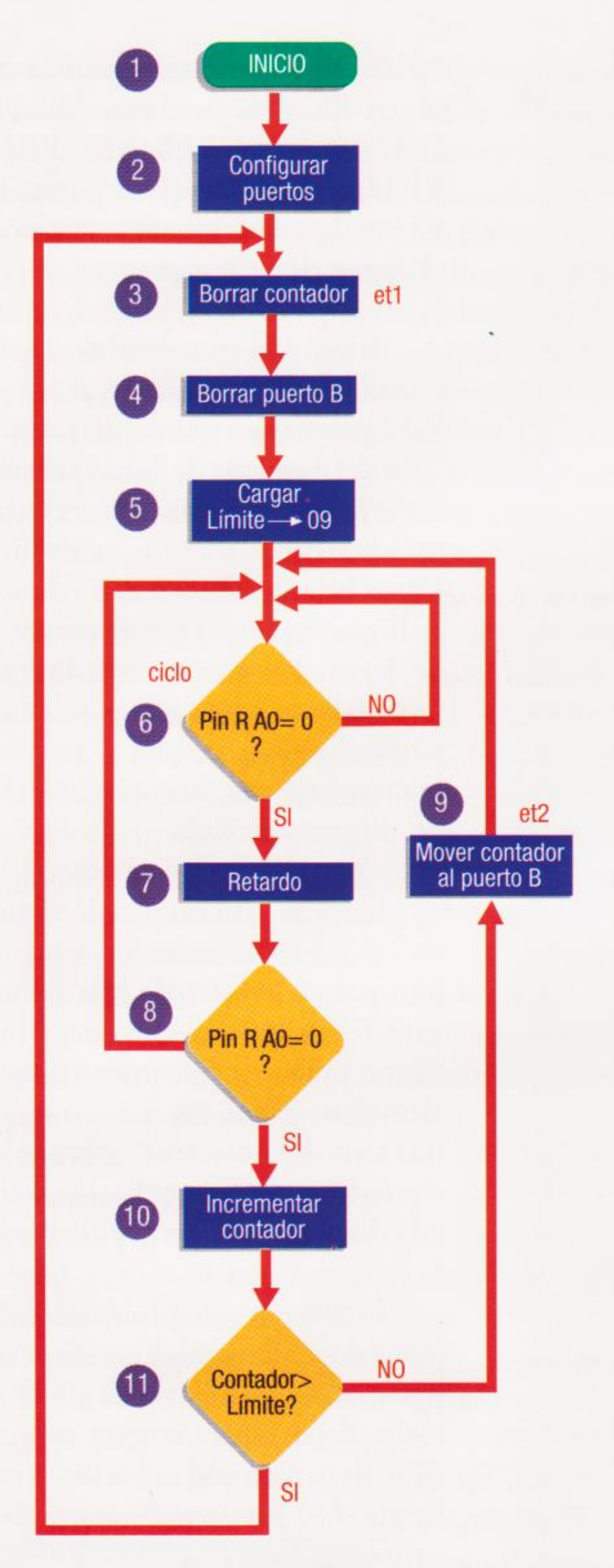


Figura 2.6 *Diagrama de flujo del programa*

rá el código objeto; en este caso sólo se le dice que se va a trabajar con un microcontrolador PIC16F84 y que el formato por defecto de los valores constantes, es el hexadecimal. En el bloque B, se asignan las direcciones en la RAM para cada uno de los registros que se van a utilizar. El bloque C del programa, utiliza la directiva ORG para establecer las direcciones del RESET y del inicio del programa, dentro de la memoria de programa del microcontrolador.

En el bloque E del código del programa y en el numeral 2 del diagrama de flujo, se establece el funcionamiento de los puertos A y B; el puerto B se configura como salida y el puerto A como entrada. Podemos notar que en esta ocasión no se utilizó el proceso tradicional de direccionar el banco 2, configurar con **trisa** y **trisb**, y luego regresar al banco 0. En este caso se utilizó la instrucción **tris**, la cual nos ahorra este trabajo al configurar directamente cada puerto con el valor que tenga el registro **w** en ese momento.

En esta misma porción del programa y en los numerales 3, 4 y 5 del diagrama de flujo, borramos el contenido de los puertos A y B y del registro **contar**, en el cual se llevará la cuenta de los pulsos originados por S1 cada vez que oprimamos y carguemos el registro **límite** con 0AH (10 en decimal); este proceso podría denominarse "inicializar los parámetros del sistema".

Cuando el conteo de pulsos ha llegado hasta nueve (9), en el *display* se mostrará un nueve, pero al siguiente pulso se debe volver a iniciar el conteo, es decir que el *display* mostrará un cero y se incrementará de nuevo con cada pulso. Antes de enviar el contenido del registro **contar** al puerto B, debemos verificar que el conteo no ha superado el número nueve; para asegurar esa situación, antes de enviar el dato al puerto B, restamos el número que vamos a enviar, al número diez.

Si esas dos cifras son iguales, el resultado de la resta dará cero, y por lo tanto la bandera de cero (Z, bit 2) del registro de estado se activará. Basado en el contenido de esa bandera, el programa puede tomar una decisión del paso que se va a seguir.

Debido a que en este proyecto el registro **contar** nunca tomará un valor superior a 10, solo utilizaremos los cuatro bits bajos (LSB) de ese registro. Por ejemplo, cuando el conteo ha llegado al número diez, **contar** tendrá el valor de **0000**

```
;PROYECTO: CONTADOR DECIMAL DE PULSOS
;CURSO PRACTICO SOBRE MICROCONTROLADORES
;CEKIT

list  p=16f84, R = HEX; especificaciones para el archvio LST


TITLE «CURSO PRACTICO SOBRE MICROCONTROLADORES»

;*****************************************************
;designación de constantes para establecer el destino
:de las operacines
;*****************************************************
w      equ    0
f      equ    1
;*****************************************************
;mapa de memoria RAM (sfr + ram de proposito general)
;*****************************************************
estado equ    03      ; registro de estado de la cpu
portb  equ    06      ; ubicación del puerto B en la RAM
porta  equ    05      ; ubicación del puerto A en la RAM
contar equ    1e      ; variable para almacenar el conteo de los
                      ; pulsos por RA0
limite equ    1d      ; limite del conteo
loops  equ    0d      ; variables auxiliares para la
loops2 equ    0e      ; rutina de retardo

reset  org    00      ; establezco el reset en la dirección 00
                      ; de la mem. de prog.
       goto   inicio  ; salto a inicio
       org    05      ; el programa se ubica desde la
                      ; posición 05 de la ROM

retardo               ;subrutina de retardo de 100 ms
       movlw  D'100'  ; el registro loops contiene el número de
       movwf  loops   ; milisegundos de esta rutina de retardo
top2   movlw  D'110'
       movwf  loops2
top    nop
       nop
       decfsz loops2  ; pregunta si terminó 1 ms
       goto   top
       decfsz loops   ; pregunta si terminó el retardo
       goto   top2
       retlw  0

inicio movlw  00      ; configuro el puerto B
       tris   portb   ; como salidas
       movlw  11      ; configuro el puerto A
       tris   porta   ; como entradas
et1    clrf   portb   ; borro el puerto B para inicializar
       clrf   porta   ; borro el puerto A
       clrf   contar  ; borro el registro de conteo de los pulsos
       movlw  0ah     ;cargo con el número oah (10) el registro
       movwf  limite  ; limite

ciclo  btfsc  porta,0 ; pregunto si el pulsador S1 está oprimido
       goto   ciclo   ; si no lo está continuo revisándolo
       call   retardo ; si está oprimido retardo 100 ms
       btfsc  porta,0 ; para comprobar
       goto   ciclo   ; si no lo está vuelve a revisar
       incf   contar,f ; si se confirma el pulsador,
                      ; incremento el contador
       movf   contar,w ; cargo el registro w con el valor
                      ; del conteo
       subwf  limite,w ; resto ese valor del valor de límite (10)
       btfss  estado,2 ; y miro si esa resta dió cero. (Z -> 1)
       goto   et2     ; si no dió cero, puedo continuar
                      ; incrementando el contador
       goto   et1     ; si dió cero, reinicio el proceso y
                      ; borro el contador
et2    movf   contar,w ; cargo w con el valor del conteo
       movwf  portb   ; llevo al puerto B dicho valor
       goto   ciclo   ; regreso a leer de nuevo el pulsador
       end
```

A B C D E F

Figura 2.7 *Código fuente del programa*

1010. Esta situación nos permite conectar directamente el decodificador 7447 a los pines RB0, RB1, RB2 y RB3 (que son los bits del puerto B) y visualizar el resultado correcto en el *display* de siete segmentos.

Ahora podemos verificar si se ha pulsado S1; esto se hace en el bloque F del programa y en los numerales 6, 7 y 8 del diagrama de flujo. Podemos notar que se hace una primera lectura del puerto RA0; si no existe nivel lógico bajo, se retorna a la etiqueta **ciclo** para que siga leyendo ese pin; si ha detectado efectivamente un nivel bajo, se llama a una ***subrutina*** de retardo y luego se chequea de nuevo si el nivel bajo permanece; numeral 8 del diagrama de flujo.

Este tipo de chequeo de seguridad es necesario en casos donde se lean pulsadores debido a la estructura física de estos dispositivos. Como ya lo mencionamos, en estos elementos es común que se origine una serie de contactos involuntarios o ***rebotes*** antes de la pulsación real y aún después de soltar el pulsador.

Para evitar este problema, además del circuito antirebote de la entrada, cuando detectamos un nivel lógico bajo esperamos un espacio corto de tiempo (del orden de los milisegundos) y leemos de nuevo para corroborar que el nivel lógico bajo se mantiene y que en realidad se trata de una entrada del pulsador y no una señal accidental.

Para incorporar el tiempo de espera entre las dos lecturas del puerto, acudimos a una ***subrutina*** de retardo.

Una ***subrutina*** es una parte especial del programa que efectúa una función específica y que puede ser llamada en cualquier momento mediante una sola instrucción. En este caso la rutina **retardo** es, como su nombre lo sugiere, un retardo de tiempo del orden de los milisegundos que utilizamos para incorporar una espera antes de calificar como válido el nivel bajo leído en el puerto A.

PARA SABER MÁS

Subrutina

Es un fragmento de instrucciones que llamamos dentro de un programa. Normalmente se coloca una subrutina dentro de un programa para una separación lógica de una función del resto del código fuente. Con este recurso podemos economizar espacio de memoria pues solo se escribe una vez el grupo de instrucciones necesarias.

La estructura de una subrutina es la siguiente:

```
Etiqueta
    Instrucción 1
    Instrucción 2
    Instrucción 3
        .
        .
    Instrucción n
    retlw
```

Etiqueta es el nombre de la subrutina el cual utilizamos para invocarla desde cualquier punto del programa a través de la directiva CALL.

Al final de cada subrutina debe aparecer la instrucción RETLW para que la CPU pueda regresar a la ejecución de las instrucciones que seguían en el programa después del CALL.

Si al hacer la segunda lectura del puerto A desaparece el nivel lógico bajo, asumimos que se trataba de un rebote, pero si se mantiene el nivel bajo, tendremos la certeza de que se trata de una pulsación en S1 e inmediatamente se incrementa el contador de pulsaciones; numeral 10 del diagrama de flujo. Como este contador no puede pasar de nueve, restamos el contenido del **contador** del contenido del registro **límite**, es decir, de 10 (numeral 11 del diagrama de flujo).

Chequeando la bandera de cero (Z) del registro de estado, nos damos cuenta si **contador** ha llegado a diez; si no ha llegado podemos mover directamente el contenido de **contador** al puerto B. Si **contador** ha llegado a diez se salta a **et1** donde se borra el contador y se reanuda el conteo.

La subrutina de retardo

Una subrutina que proporcione un tiempo de retardo, es muy útil en una gran cantidad de aplicaciones con microcontroladores. En el bloque D del código fuente, observamos la subrutina de retardo desarrollada la cual pasamos a analizar: "**retardo**" es el nombre o etiqueta que indica el inicio de las instrucciones de la subrutina que será llamada desde el programa principal.

En este caso, el retardo de tiempo se logra ejecutando miles (o millones) de instrucciones que en realidad no hacen nada.

Como la ejecución de una instrucción emplea cierto tiempo, dependiendo del procesador y del circuito oscilador que posea, al introducirnos en un ciclo que consuma miles de instrucciones y que no pueda continuar con el programa hasta que esa cantidad sea ejecutada, obtendremos un espacio de tiempo que nos será útil para nuestro programa. Este tipo de retardo se denomina retardo por *software* y lo único que hace es que, cuando invocamos la subrutina, se entra en un período de espera que puede ajustarse según las necesidades del sistema.

Observamos en el programa que el registro **loops** se carga con el número decimal 100 y el registro **loops2** con el número decimal 110. El retardo se inicia en la subrutina **top2**; allí se empieza a decrementar el registro **loops2** y hasta que no llegue a cero, el programa permanece haciendo ese ciclo.

Dentro de ese ciclo se recorren unas instrucciones NOP (no operación), que simplemente no hacen nada, solo ayudan en la rutina a incrementar el número de instrucciones que se deben ejecutar.

Cuando **loops2** ha llegado a cero, se decrementa en una unidad el registro **loops** y se carga nuevamente **loops2** con su valor original.

De esta manera, logramos que se ejecute 100 veces el decremento total de **loops2**. Hasta que **loops** no sea igual a cero, la rutina se continuará ejecutando y no se retornará al programa principal.

Ensamblaje del proyecto

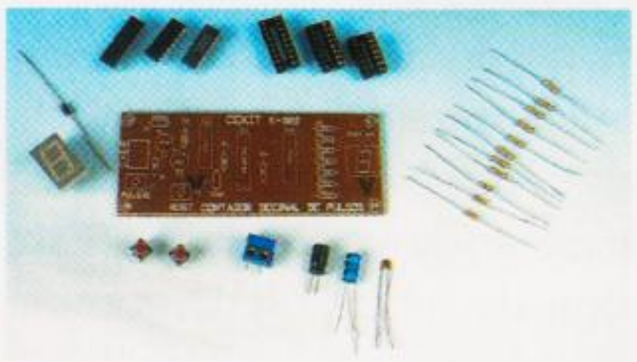


Figura 2.8 *Aquí se muestra el conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el proyecto.*

LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F84 programado
1	IC2	Decodificador de BCD a 7 segmentos para ánodo común, 7447
1	IC3	Compuertas inversoras *schmitt trigger*, 7414
1	D1	Diodo rectificador de propósito general 1N4001 o 1N4004
1	DISPLAY1	*Display* de 7 segmentos ánodo común
1	C1	Condensador electrolítico de 47uF - 16V
1	C2	Condensador electrolítico de 10uF - 16V
1	C3	Condensador cerámico de 20pF - 50V
2	R1 y R4	Resistencias de 1 KΩ - 1/4W
8	R2,R5,R6,R7,R8,R9,R10,R11	Resistencias de 330 Ω - 1/4W
1	R3	Resistencia de 10 KΩ -1/4W
2	S1 y S2	Pulsador pequeño de 4 pines, para circuito impreso
1	K-322	Circuito impreso CEKIT
1		Base para circuito integrado de 18 pines
1		Base para circuito integrado de 14 pines
1		Base para circuito integrado de 16 pines
1		Conector de tornillo de 2 pines

Paso 1

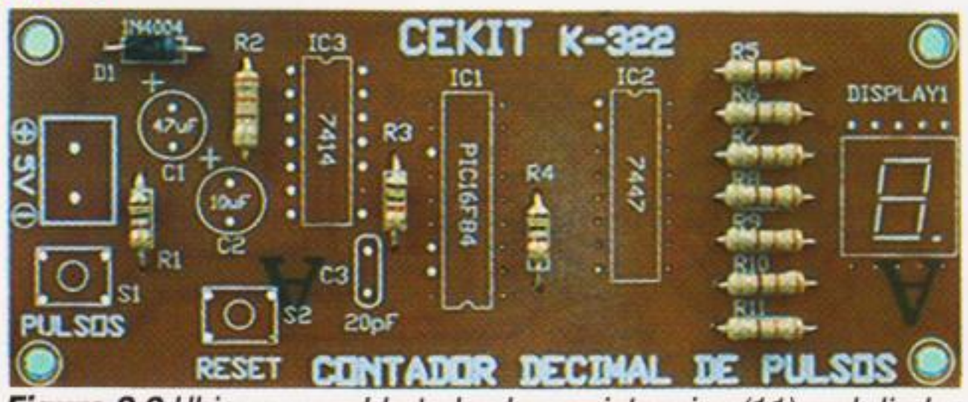


Figura 2.9 *Ubique y suelde todas las resistencias (11) y el diodo rectificador D1.*

Paso 2

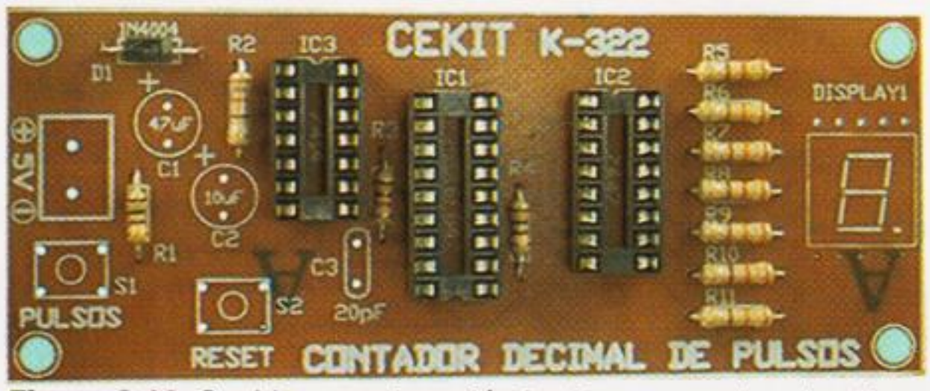


Figura 2.10 *Suelde a continuación las bases para los circuitos integrados.*

Paso 3

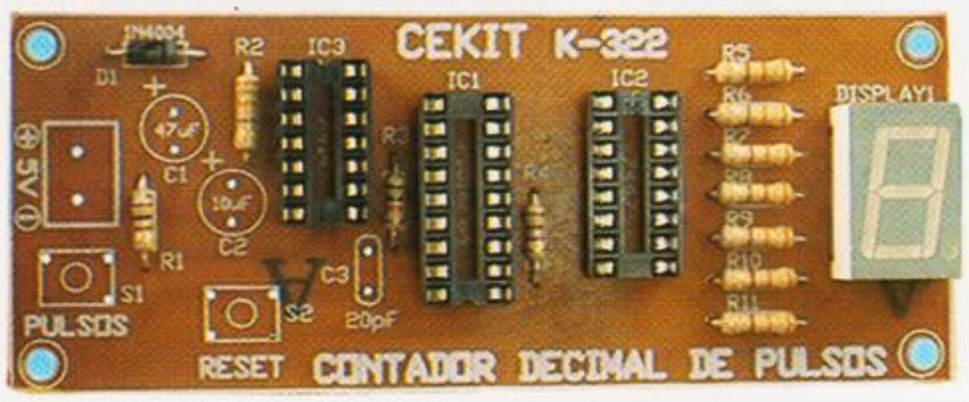


Figura 2.11 *Instale y suelde ahora el display de siete segmentos. Observe la posición correcta del punto decimal, el cual debe quedar abajo a la derecha.*

Paso 4

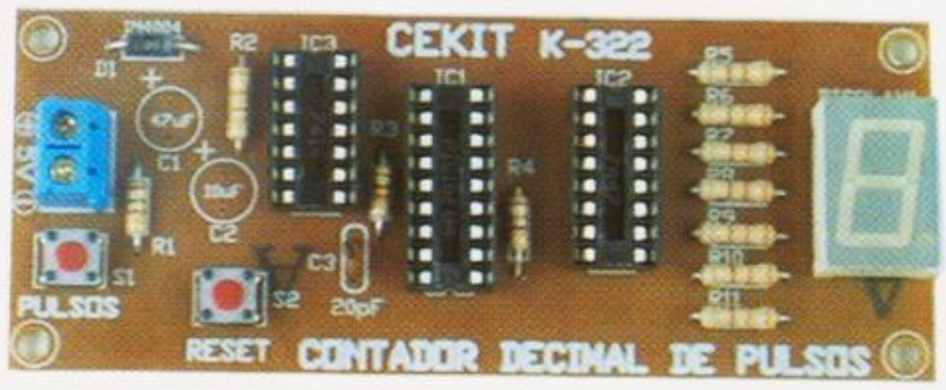


Figura 2.12 *Instale y suelde los dos interruptores pulsadores y el terminal para la alimentación.*

Paso 5

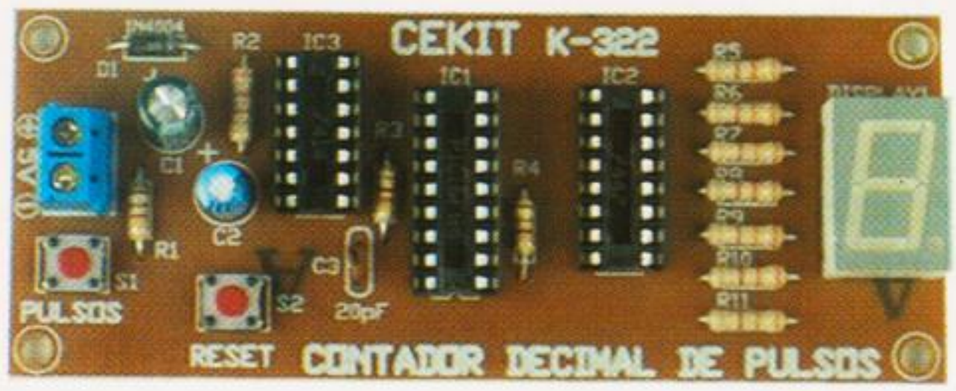


Figura 2.13 *Instale ahora, con soldadura, el condensador cerámico primero, y luego los dos condensadores electrolíticos observando su polaridad.*

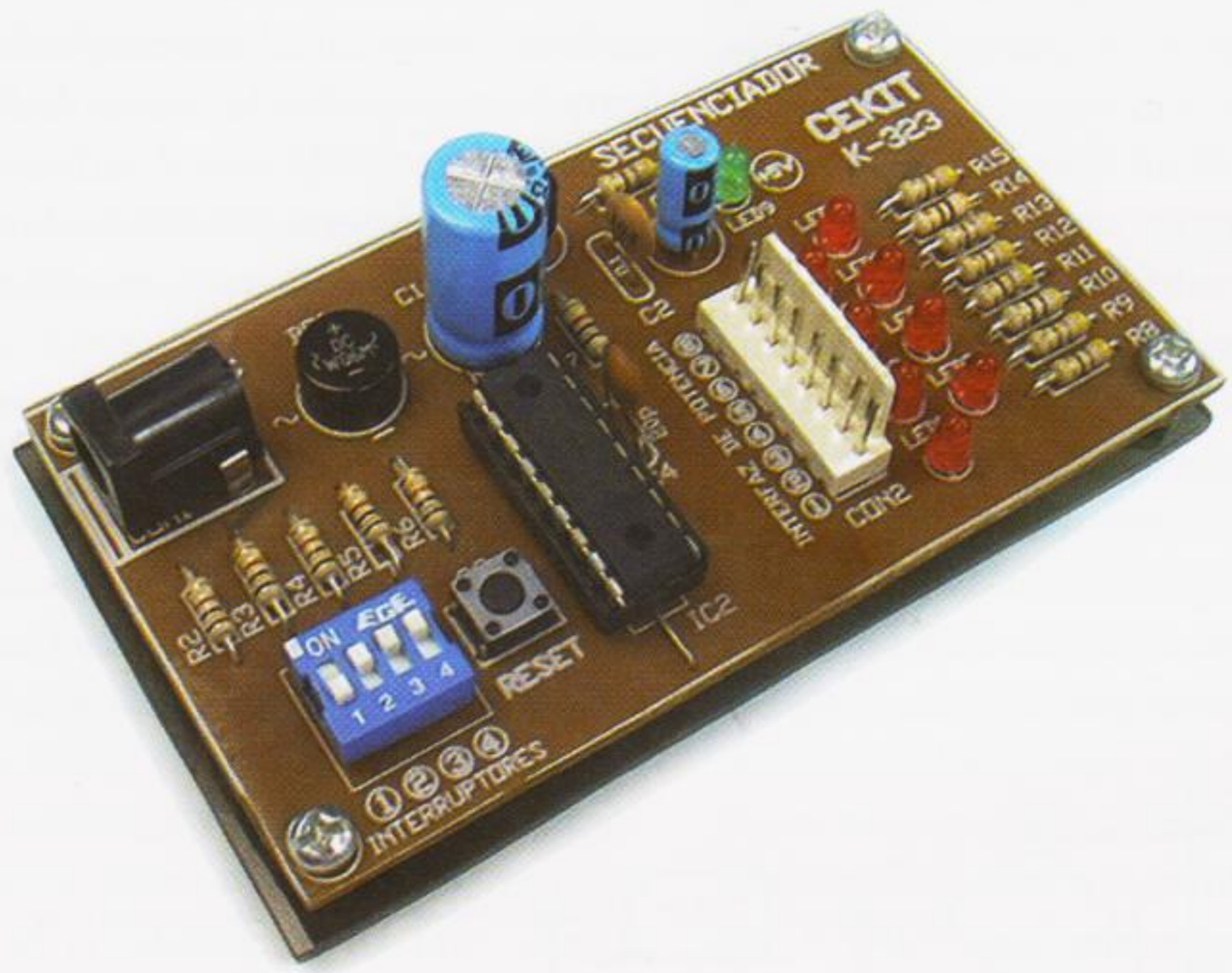
Paso 6

Figura 2.14 *Instale en su base el microcontrolador PIC 16F84 programado y los circuitos integrados 7447 y 7414 observando para cada uno, la posición correcta del pin No.1*

Paso 7. Prueba final

Para probar el circuito, aliméntelo con una fuente regulada de 5 voltios y presione el pulsador RESET; el display debe indicar 0. Presione luego el interruptor S1; observe que cada vez que se presiona, el display debe incrementar el conteo en 1. Si esto ocurre, el circuito está trabajando bien. Si no es así, revise muy bien la posición de los componentes, todas las soldaduras y el programa del microcontrolador.

PROYECTO 3



SECUENCIADOR DE 8 CANALES

En esta ocasión construiremos un dispositivo que origina secuencialmente una serie de datos de 8 *bits*. Este circuito puede usarse en juegos de luces o en el control secuencial de algún proceso o aparato. La salida se visualiza por medio de diodos LED, pero puede adaptarse a una tarjeta de potencia con tiristores para activar bombillas de mayor voltaje.

Características

- Secuenciador de ocho *bits* u ocho salidas
- Configuración de la velocidad con que se origina la secuencia: cuatro posibilidades
- Configuración de la secuencia: ocho posibilidades
- Cada secuencia es de ocho pasos
- Salida por medio de diodos LED e interfaz hacia circuitos de potencia

Funcionamiento

Un secuenciador de luces es uno de los proyectos que todo aficionado a la electrónica ha querido fabricar alguna vez. Por esta razón y por la utilidad práctica que puede tener un dispositivo de esta naturaleza, hemos querido presentarles este circuito, el cual se puede ampliar o acomodar según sus necesidades. Un secuenciador, como su nombre lo sugiere, es un aparato que a través de sus salidas puede enviar varios datos digitales uno tras otro, cumpliendo con una secuencia establecida. Por lo general, una secuencia se compone de un número determinado de datos almacenados en una tabla que constituyen los pasos de la secuencia. El proceso empieza tomando el primer dato de la tabla y llevándolo al puerto de salida, donde permanece por un tiempo determinado. Luego se repite el proceso con el segundo dato, luego con el tercero y así sucesivamente hasta recorrer todos los datos de la tabla. Cuando se ha terminado con el último dato, se llama otra vez al primero, volviendo a iniciar el proceso en una especie de ciclo continuo.

Figura 3.1 *Diagrama de bloques del secuenciador*

Este secuenciador permite por medio de un interruptor tipo *dipswitch*, seleccionar la velocidad con que se envían los datos secuenciales al puerto, es decir, el tiempo que cada uno de ellos va a permanecer en la salida. De igual manera permite seleccionar la secuencia o tabla que se va a trabajar,

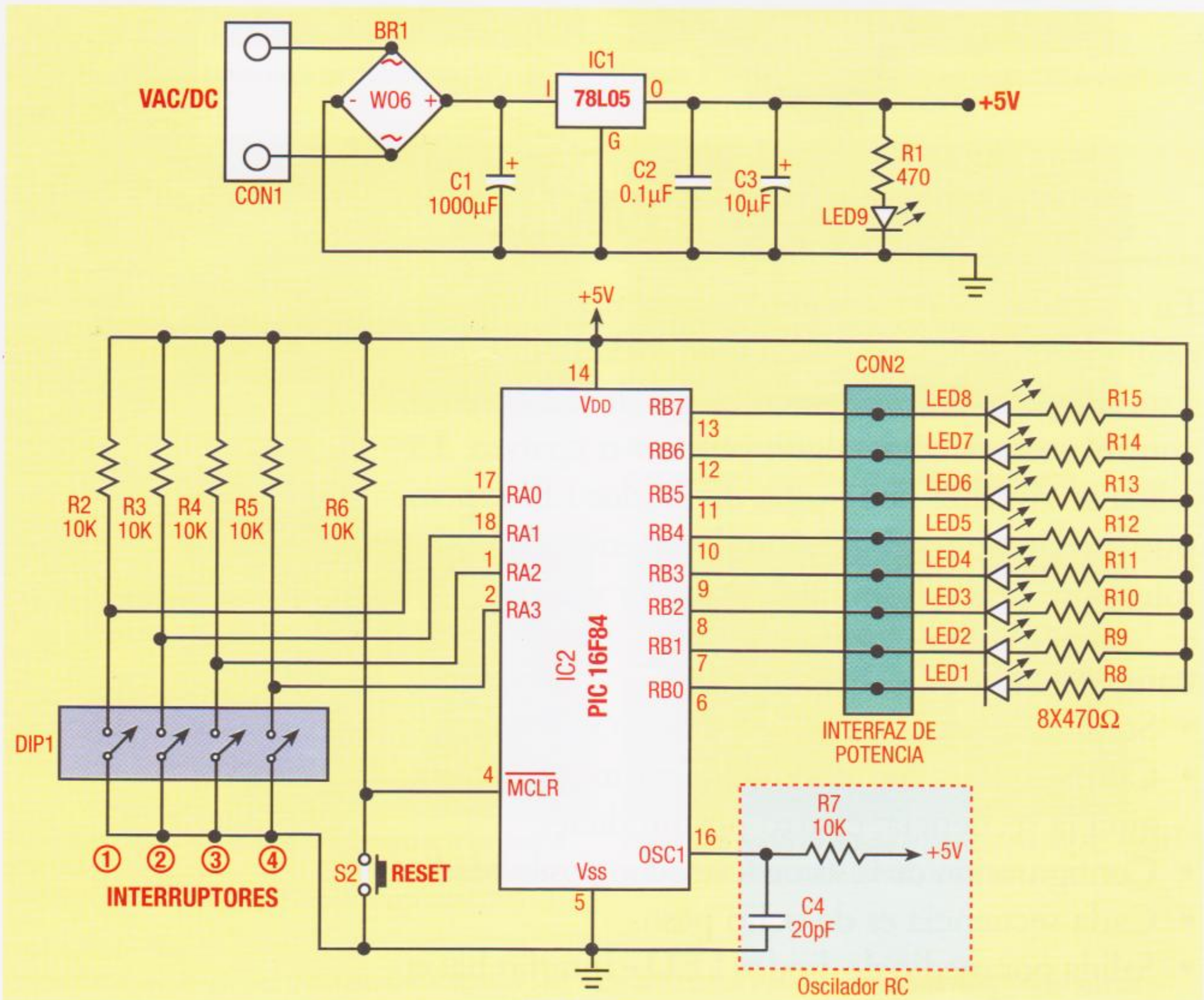


Figura 3.2 *Diagrama esquemático del secuenciador*

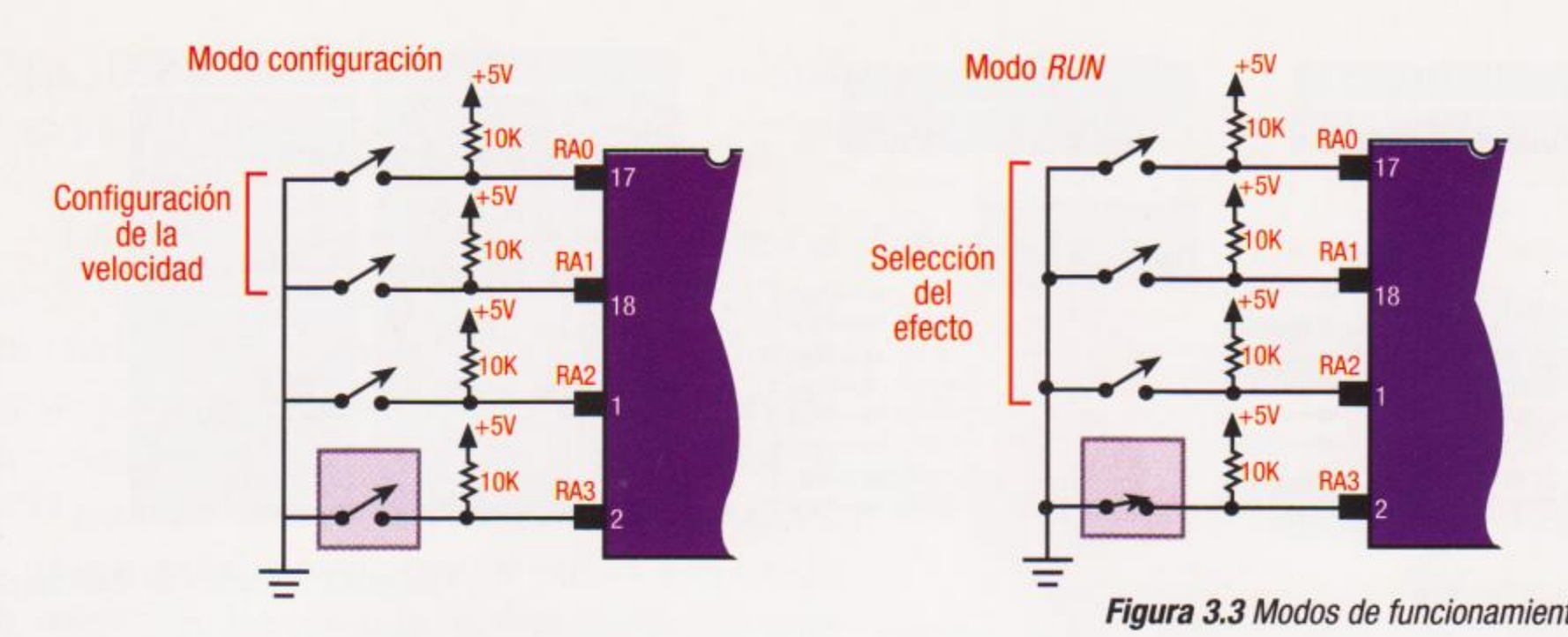


Figura 3.3 *Modos de funcionamiento del sistema*

entre ocho posibilidades que se han establecido dentro del programa. En otras palabras: existen ocho tablas distintas, cada una de ocho pasos y ocho *bits*, entre las cuales se escoge una para establecer la secuencia. En la **figura 3.1** tenemos el diagrama de bloques general del proyecto, en el se observa al *dipswitch* como dispositivo de entrada, un microcontrolador PIC16F84 como elemento de procesamiento y ocho diodos LED como elementos de salida.

En la **figura 3.2** se ilustra el diagrama esquemático del circuito. El *dipswitch* es de cuatro pines y se ha conectado a los terminales RA0, RA1, RA2 y RA3 del microcontrolador, el puerto B se ha utilizado como salida de la secuencia; por lo tanto, allí van conectados los ocho diodos LED cada uno de los cuales se activa con un nivel **bajo** en la salida del puerto. Como oscilador se ha utilizado una red RC y para la alimentación se utilizó un regulador 78L05 con un circuito de filtrado.

Configuración del sistema

Podemos seleccionar entre cuatro velocidades y ocho tipos de secuencias distintas, todo con un *dipswitch* de cuatro pines. Este sistema se ha diseñado para que funcione en dos formas o modos: el modo "programación" y el modo normal o "RUN". La elección del modo de trabajo se hace a través del pin 4 del *dipswitch* conectado al pin RA3; si este interruptor se encuentra abierto, en RA3 se leerá un nivel **alto** y el sistema entenderá que está en modo "programación"; y si el interruptor se encuentra cerrado, en RA3 se leerá un nivel **bajo** y el sistema entrará en modo "RUN", **figura 3.3**.

Cuando estamos en el modo de programación, podemos establecer la velocidad del secuenciador leyendo los *dipswitch* 1 y 2 conectados a los pines RA0 y RA1, teniendo en cuenta la **Tabla 3.1**. En el momento que activamos el pin 4 del *dipswitch* en RA3 se lee un nivel **bajo** y entramos a modo RUN. La velocidad seleccionada ya se ha almacenado y ahora podemos seleccionar la secuencia o el efecto que se visualizará en los diodos LED. La selección de la secuencia depende del dato originado por los pines 1, 2 y 3 del *dipswitch* conectados a los terminales RA0, RA1 y RA2, con los que se selecciona una entre las ocho tablas que se han construido.

En la **figura 3.4** se muestran los dos primeros efectos que se originan cuando se han seleccionado las tablas 1 y 2. Cuando nos encontremos en modo RUN podemos cambiar en cualquier momento el tipo de secuencia a enviar por el puerto, pero no podremos cambiar la velocidad; si se desea cambiar la velocidad, se debe abrir el pin 4 del *dipswitch* y darle un *reset* al sistema por medio del pulsador S2. En ese momento, entramos de nuevo al modo de programación.

Descripción del programa

Antes de analizar en detalle el programa, analicemos como trabajar con tablas en lenguaje ensamblador, lo cual es una herramienta de gran utilidad en el

RA0	RA1	Velocidad (Aprox.)
0	0	0.3 seg.
1	0	0.5 seg.
0	1	0.7 seg.
1	1	0.9 seg.

Tabla 3.1 *Configuración de la velocidad*

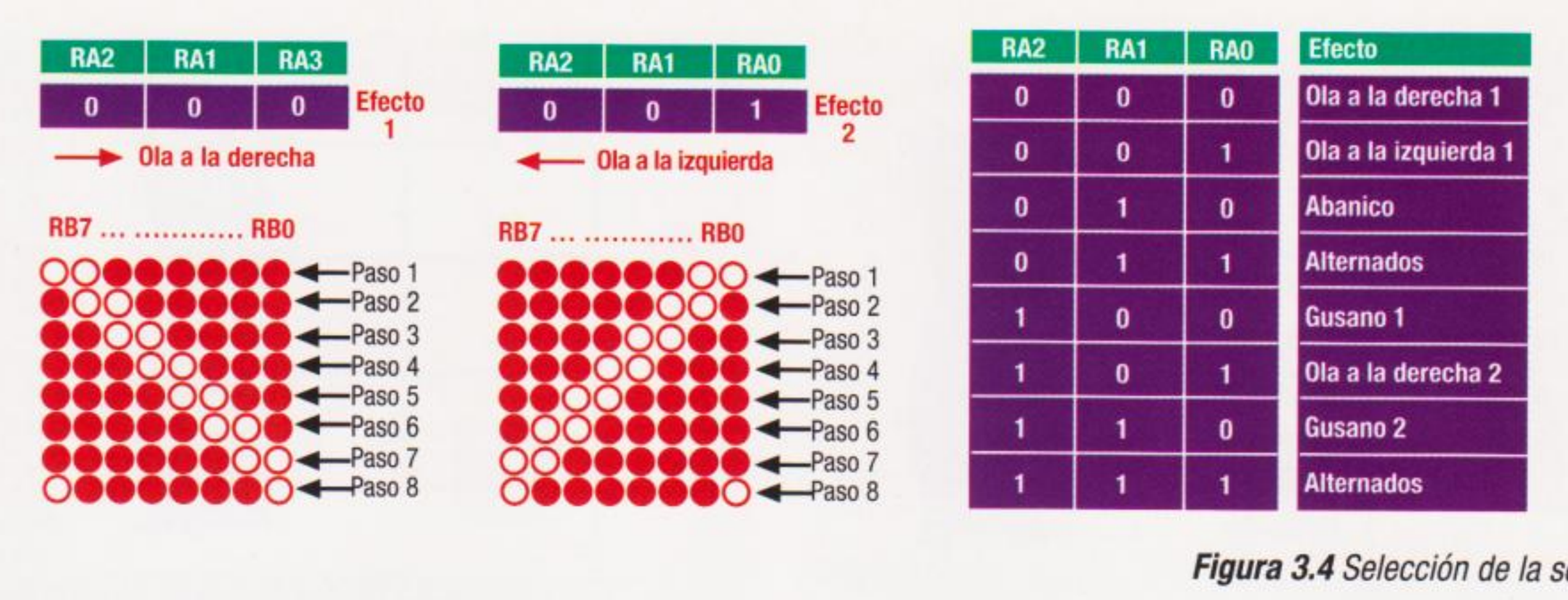


Figura 3.4 *Selección de la secuencia*

desarrollo de aplicaciones y que se utilizará en repetidas ocasiones en este proyecto.

Existen varios métodos para acceder a las posiciones de memoria denominados *métodos de direccionamiento*. Uno de ellos corresponde al *direccionamiento relativo* el cual básicamente altera el contenido del contador de programa (PC). De esta manera se logra saltar a alguna posición desde el punto donde se encuentra el PC hasta una posición determinada. En la **figura 3.5** observamos la estructura básica del manejo de una tabla localizada dentro de la memoria de programa.

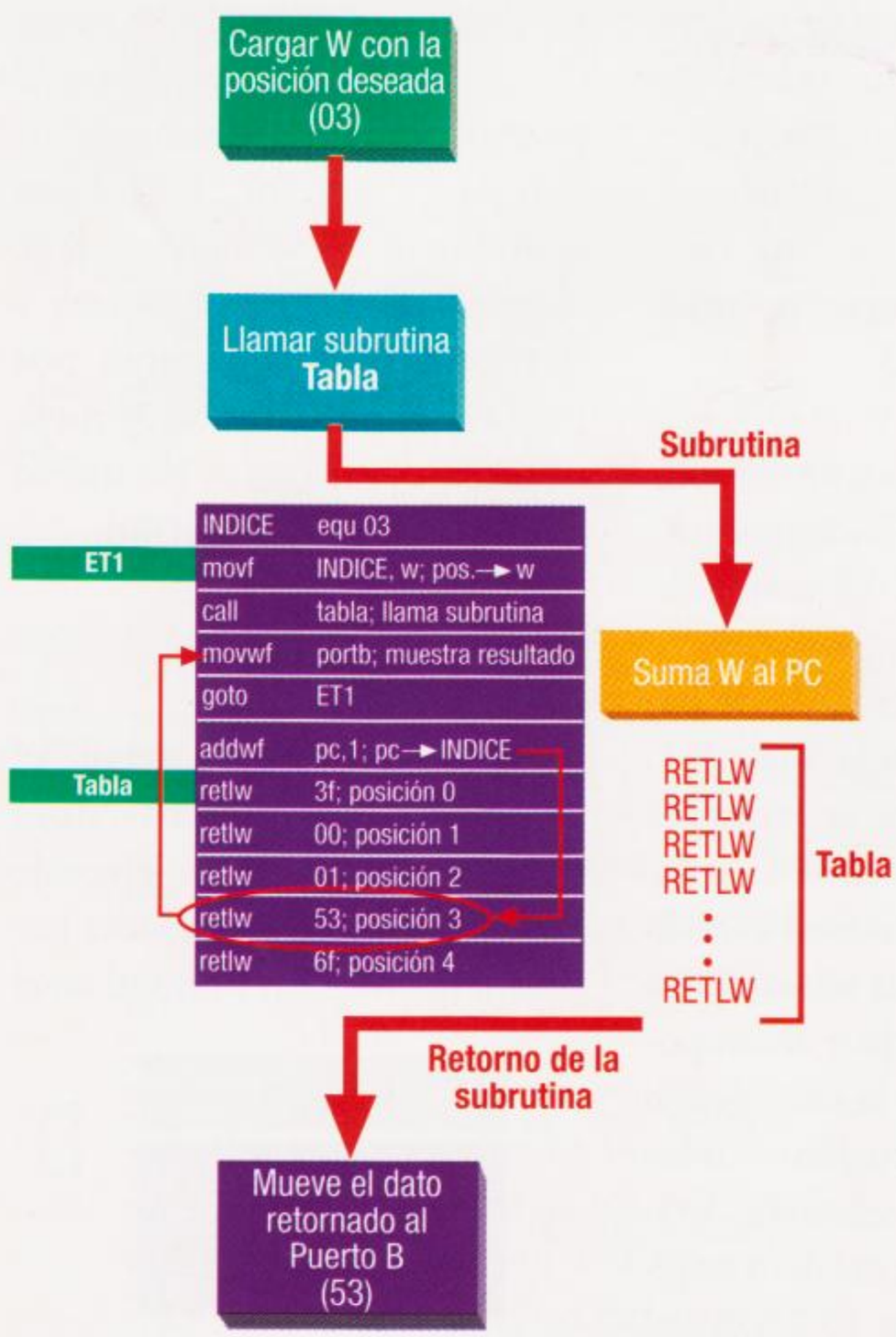


Figura 3.5 *Manejo de tablas*

Inicialmente se carga el registro de trabajo (W) con el valor índice o la posición de la tabla que queremos acceder; luego se llama la subrutina TABLA en donde al PC se le suma el contenido de W y por lo tanto la CPU se dirige a la posición en la tabla dada por W (03 en este ejemplo). Cada una de las posiciones de la tabla tiene la instrucción RETLW, la cual causa que el registro W se cargue con el valor ubicado a continuación en la misma línea y se salga de la subrutina. Cuando se regresa al flujo normal del programa, W viene cargado con el nuevo dato tomado de la tabla (53 en este ejemplo) y ese nuevo dato es el que se envía por el puerto.

En la **figura 3.6** observamos el diagrama de flujo del programa y en la **figura 3.7** el código fuente del mismo. En el **bloque A** del programa se establece que se va a trabajar con un PIC16F84; que el formato por defecto es el hexadecimal, se asignan las posiciones de las variables en memoria RAM y se establecen las direcciones de RESET y del inicio del programa, dentro de la memoria de programa del microcontrolador. La forma como se hacen estas asignaciones ya se explicó en los dos proyectos anteriores. En el **bloque C** del código del programa y en el numeral 2 del diagrama de flujo, se establece el funcionamiento de los puertos A y B; el puerto B se configura como salida y el puerto A se configura como entrada. De igual manera, en esta porción del código se limpia el contenido de los registros VALOR y PORTB para no correr el riesgo de que

algún dato parásito o aleatorio se cargue en estas posiciones de memoria.

En el **bloque D** y numeral 3, se selecciona el modo de operación del sistema; si el pin RA3 lee un nivel **alto** entra al modo de programación y si lee un nivel **bajo**, entra al modo RUN. En el modo de programación, **bloque E**, el puerto B se pone en 00H para que todos lo diodos LED se enciendan y el usuario pueda asegurarse que está en este modo de operación. En el modo "programación" se establece la velocidad con que salen al puerto los datos de la secuencia dependiendo de la lectura de los pines RA0 y RA1 del puerto A. Se lee el puerto A y ese valor se lleva a una operación lógica AND con el número 03H (**0000 0011**) el cual se ha almacenado en el registro TEMPORAL; esta operación se hace para enmascarar o permitir que sólo queden los *bits* RA0 y RA1 con el valor leído del puerto, los demás *bits* quedan en cero; numerales 5 y 6 del diagrama de flujo. Finalmente el valor enmascarado es llevado al registro VALOR, quien configura la velocidad de la secuencia, numeral 7.

Después de que se ha cargado VALOR, se retorna a la etiqueta FUNCION para determinar si en el pin RA3 ya se ha dado la señal de habilitación del modo RUN; mientras esto no ocurra, el programa entra en un ciclo de lectura y enmascaramiento del puerto A, para determinar el valor definitivo del registro VALOR. Cuando finalmente se ha detectado un nivel **bajo** en RA3, entramos al modo RUN, **bloque F** del código. Ahora leemos de nuevo el puerto A y esa lectura la movemos al registro TDT (Tipo De Tabla) en el cual se enmascaran los tres primeros *bits* que corresponden a RA0, RA1 y RA2, numerales 9 y 10. El valor almacenado en TDT servirá para llamar la tabla correspondiente a cada tipo de efecto, numeral 11.

En el **bloque G** es donde se establece cual tabla se va a llamar entre las 8 posibles. Por ejemplo, para determinar si la lectura fue uno, al valor cargado en TDT se le resta 01H, si esta resta da cero, significa que en TDT hay 01H también, y si eso ocurre, la

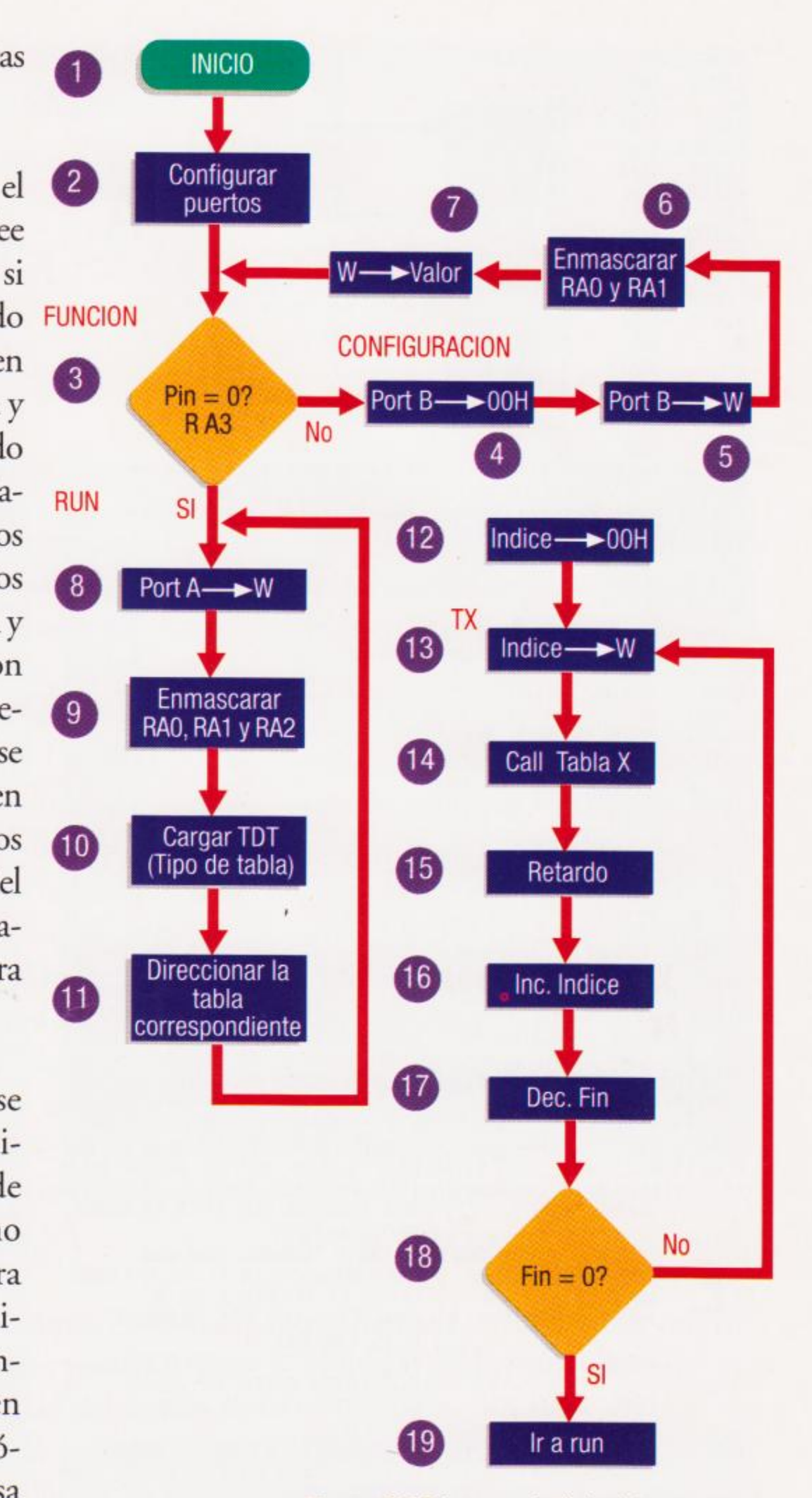


Figura 3.6 *Diagrama de flujo del programa*

bandera Z del registro de estado (STATUS REGISTER) se pondrá en nivel **alto**. Por eso se puede utilizar esa bandera para verificar la igualdad. Este proceso se repite para cada valor entre cero y ocho, y en el momento que se determine el valor exacto de TDT se llamará a la tabla correspondiente.

Cuando se ha determinado cual es la tabla correcta, se llama la rutina correspndiente en el **bloque H** (TABLA_x), numeral 14. El registro INDICE, que como su nombre lo indica, es el

A

```
PROCESSOR        16F84
RADIX   HEX
PC      EQU     02
FIN     EQU     1E
INDICE  EQU     2D
LAZ01   EQU     13
VALOR   EQU     14
LAZ     EQU     10
LAZOQ   EQU     11
ESTADO  EQU     03
TEMPORAL EQU    15
PORTA   EQU     05
PORTB   EQU     06
W       EQU     0
F       EQU     1
TDT     EQU     12
RESET   ORG     0
        GOTO    INICIO
        ORG     5
;*****************************************************
;RUTINA DE RERTARDO, DEPENDE DEL REGISTRO VALOR
;*****************************************************
RETARDOMOVF     VALOR,W
        ADDLW   01
        MOVWF   LAZ
RETAR   MOVLW   D'70'
        MOVWF   LAZ01
        MOVF    VALOR,W
        ADDWF   LAZ01,F
R11     MOVLW   D'110'
        MOVWF   LAZOQ
        MOVF    VALOR,W
        ADDWF   LAZOQ,F
R1
        DECFSZ  LAZOQ
        GOTO    R1
        DECFSZ  LAZ01
        GOTO    R11
        DECFSZ  LAZ
        GOTO    RETAR
        RETLW   0
```

B

```
INICIO  MOVLW   B'00000000' ;CONFIGURA EL PUERTO B COMO SALIDA
        TRIS    PORTB       ;ALLI VAN LOS LED
        MOVLW   B'11111111' ; Y EL PUERTO A COMO ENTRADAS
        TRIS    PORTA       ; ALLI VA EL DIPSWITCH
        CLRF    PORTB       ;INICIALIZO EL PUERTO B CON CEROS
        CLRF    VALOR       ; INICIALIZO LA VELOCIDAD
FUNCION:        ; SELECCION ENTRE MODO PROGRAMACION/RUN
        BTFSC   PORTA,3     ;  SELECCION DE FUNCION
        GOTO    CONFIGURACION ; PROGRAM. DE LA VELOCIDAD
        GOTO    RUN         ; MODO RUN
```

C

D

```
CONFIGURACION: ; AQUI ESTABLEZCO LA VELOCIDAD DEL SECUENCIADOR
        MOVLW   B'00000000' ; SE ENCIENDE TODO EL PUERTO B
        MOVWF   PORTB     ; PARA INDICAR QUE ESTA EN MODO
                          √ PROGRAMACION
        MOVLW   B'00000011' ; CARGO TEMPORAL CON 03H
        MOVWF   TEMPORAL  ; PARA ENMASCARAR EL DATO LEIDO
        MOVF    PORTA,W   ; CARGO EL PUERTO A EN W
        ANDWF   TEMPORAL,W; ENMASCARO LOS DOS PRIMEROS
                          ; BITS RA0-RA1
        MOVWF   VALOR     ; LA LECTURA DE LA VELOCIDAD
                          ; REQUERIDA QUEDA EN VALOR
        GOTO    FUNCION   ; REGRESO A LEER EL DIP4
```

E

```
RUN:    MOVLW   07H       ; CADA EFECTO ES DE 8 PASOS
        MOVWF   FIN       ; FIN ES QUIEN DA EL LIMITE PARA
                          ; EL INDEXADO
        CLRF    TDT       ; INICIALIZO ESTE REGISTRO
        MOVF    PORTA,W   ; EL PUERTO A LO LLEVO A W
        MOVWF   TDT       ; SELECCIONAR LA TABLA INDICADA
        BCF     TDT,3     ; ENMASCARO LOS BITS QUE
                          ; NECESITAMOS
        BCF     TDT,4     ; RA0, RA1 Y RA2
        MOVF    TDT,W
        SUBLW   00H       ; DETERMINAR SI SE HA LEIDO CERO
                          ; EN EL DIP
        BTFSC   ESTADO,2  ; SI SE LEE CERO, ENTONCES
        GOTO    TABLA_1   ; SALTAMOS A LA TABLA 1
```

F

```
        MOVF    TDT,W
        SUBLW   01H       ;  HA LEIDO UNO EN EL DIP?
        BTFSC   ESTADO,2  ; SI SE LEE UNO, ENTONCES
        GOTO    TABLA_2   ; SALTAMOS A LA TABLA 2
        MOVF    TDT,W
        SUBLW   02H       ;  HA LEIDO DOS EN EL DIP ?
        BTFSC   ESTADO,2  ; SI SE LEE DOS, ENTONCES
        GOTO    TABLA_3   ; SALTAMOS A LA TABLA 3
        MOVF    TDT,W
        SUBLW   03H       ;  HA LEIDO TRES EN EL DIP?
        BTFSC   ESTADO,2  ; SI SE LEE TRES, ENTONCES
        GOTO    TABLA_4   ; SALTAMOS A LA TABLA 4
        MOVF    TDT,W
        SUBLW   04H       ; HA LEIDO CUATRO EN EL DIP?
        BTFSC   ESTADO,2  ; SI SE LEE CUATRO, ENTONCES
        GOTO    TABLA_5   ; SALTAMOS A LA TABLA 5
```

G

```
        MOVF    TDT,W
        SUBLW   05H       ;  HA LEIDO CINCO EN EL DIP?
        BTFSC   ESTADO,2  ; SI SE LEE CINCO, ENTONCES
        GOTO    TABLA_6   ; SALTAMOS A LA TABLA 6
        MOVF    TDT,W
        SUBLW   06H       ; HA LEIDO SEIS EN EL DIP?
        BTFSC   ESTADO,2  ; SI SE LEE SEIS, ENTONCES
        GOTO    TABLA_7   ; SALTAMOS A LA TABLA 7
        MOVF    TDT,W
        SUBLW   07H       ;  HA LEIDO SIETE EN EL DIP?
        BTFSC   ESTADO,2  ; SI SE LEE SIETE, ENTONCES
        GOTO    TABLA_8   ; SALTAMOS A LA TABLA 8
TABLA_1MOVLW    00H       ; CARGAMOS INDICE CON LA PRIMERA
        MOVWF   INDICE    ; POSICION DE LA TABLA
T1      MOVF    INDICE,W  ; INDEXAMOS LA POSICION EN LA
                          ; TABLA
        CALL    TABLA1    ; LLAMAMOS LA TABLA 1
        MOVWF   PORTB     ; LA LECTURA LA MANDAMOS AL
                          ; PUERTO B
        CALL    RETARDO   ; Y SE DEJA DURANTE UN TIEMPO
        INCF    INDICE,F  ; SE INCREMENTA EL INDICE DE LA
                          ; TABLA
        DECFSZ  FIN       ; Y SE DECREMENTA EL REGISTRO
                          ; LIMITE
        GOTO    T1        ; SI NO HA LLEGADO A CERO SE
                          ; REPITE EL PROCESO
        GOTO    RUN       ; SI LIMITE LLEGA A CERO SALIMOS
                          ; DE LA RUTINA
TABLA1  ADDWF   PC,1      ; ESTA ES LA TABLA 1, DONDE SE
        RETLW   B'11000000' ; REGISTRAN LOS VALORES QUE SE
        RETLW   B'01100000' ; MANDARAN SECUENCIALMENTE POR
        RETLW   B'00110000' ; EL PUERTO B
        RETLW   B'00011000'
        RETLW   B'00001100'
        RETLW   B'00000110'
        RETLW   B'00000011'
        RETLW   B'10000001'
```

H

```
TABLA_8 MOVLW   00H       ; CARGAMOS INDICE CON LA PRIMERA
        MOVWF   INDICE    ; POSICION DE LA TABLA
T8      MOVF    INDICE,W  ; INDEXAMOS LA POSICION EN LA
                          ; TABLA
        CALL    TABLA8    ; LLAMAMOS LA TABLA 3
        MOVWF   PORTB     ; LA LECTURA LA MANDAMOS AL
                          ; PUERTO B
        CALL    RETARDO   ; Y SE DEJA DURANTE UN TIEMPO
        INCF    INDICE,F  ; SE INCREMENTA EL INDICE DE LA
                          ; TABLA
        DECFSZ  FIN       ; Y SE DECREMENTA EL REGISTRO
                          ; LIMITE
        GOTO    T8        ; SI NO HA LLEGADO A CERO SE
                          ; REPITE EL PROCESO
        GOTO    RUN       ; SI LIMITE LLEGA A CERO SALIMOS
                          ; DE LA RUTINA
TABLA8  ADDWF   PC,1      ; ESTA ES LA TABLA 8, DONDE SE
        RETLW   B'11100111'; REGISTRAN LOS VALORES QUE SE
        RETLW   B'11000011' ; MANDARAN SECUENCIALMENTE POR
        RETLW   B'10000001' ; EL PUERTO B
        RETLW   B'00000000'
        RETLW   B'10000001'
        RETLW   B'11000011'
        RETLW   B'11100111'
        RETLW   B'11111111'

        END
```

Figura 3.7 *Código fuente del proyecto*

índice que nos va a servir para recorrer toda la tabla. Este INDICE se establece inicialmente en 00H que corresponde a la primera posición en la tabla. El contenido de este registro se lleva a W y luego se llama a la rutina TABLA que es donde se ubican los datos de la secuencia. Cuando en W se ha recuperado el dato de la tabla indicado por INDICE, este dato se envía al puerto B para que se visualice en los diodos LED; enseguida se invoca la rutina RETARDO, la cual permite que el dato permanezca en el puerto durante determinado tiempo. Después de cumplido el tiempo de retardo, se incrementa el registro INDICE para direccionar la siguiente posición en la tabla y se decrementa el registro FIN, numerales 16, 17 y 18 del diagrama de flujo. Este último registro es una especie de contador iniciado con 07H en el bloque F, que impide que INDICE direccione posiciones inexistentes en la tabla, esto es, como cada una de las tablas solo tiene ocho posiciones, INDICE no debe direccionar por ejemplo, la posición diez de la tabla; si se da este caso, se presentará un error.

De esta manera INDICE sólo se incrementará siete veces, mientras FIN llega a cero. Tenga en cuenta que inicialmente INDICE toma el valor de 00H; con este valor se direcciona la primera posición de la tabla y por lo tanto sólo debe ser incrementado siete veces para hacer el barrido completo.

La rutina de retardo, **bloque B**, produce un retardo por *software* directamente proporcional al contenido del registro VALOR, el cual recordemos, se estableció en la fase de programación. La dependencia consiste simplemente en que al valor de los registros que se decrementan, se le adiciona el contenido del registro VALOR. De esta manera el tiempo que permanece un dato en el puerto depende de la configuración inicial.

Cada uno de los datos de la tabla seleccionada se van sacando uno a uno por el puerto B, cuando ha salido el último, el programa retorna a la etiqueta RUN, bloque F, para reanudar el proceso y leer de nuevo la secuencia seleccionada, la cual pudo haber sufrido algún cambio.

Nota: *Por razones de espacio en muchas ocasiones no podremos publicar el código completo de los programas. Todos ellos estan disponibles en nuestra página: www.cekit.com.co.*

El *byte* de potencia

El circuito puede manejar cargas de mayor potencia a través de una interfaz que hemos denominado el "*byte* de potencia". El circuito con referencia K-79 de CEKIT, es una interfaz optoacoplada con triacs, que se utiliza para efectuar la conmutación segura de cargas CA que se alimentan con 110V o 220V y consumen hasta 10A. El circuito se ha diseñado con ocho módulos iguales, tal como se muestra en el diagrama esquemático simplificado de la **figura 3.8**. Cada módulo tiene a la entrada de CC un optotriac (MOC 3021) que se encarga de aislar los niveles de tensión de la carga y del circuito de control, y activar por medio de la luz emitida por un diodo LED, un pequeño triac de disparo óptico. Éste, a su vez, hace disparar el triac de potencia y la carga conectada a él. Las salidas de los triacs se conectan a diferentes circuitos con bombillos o lámparas incandescentes.

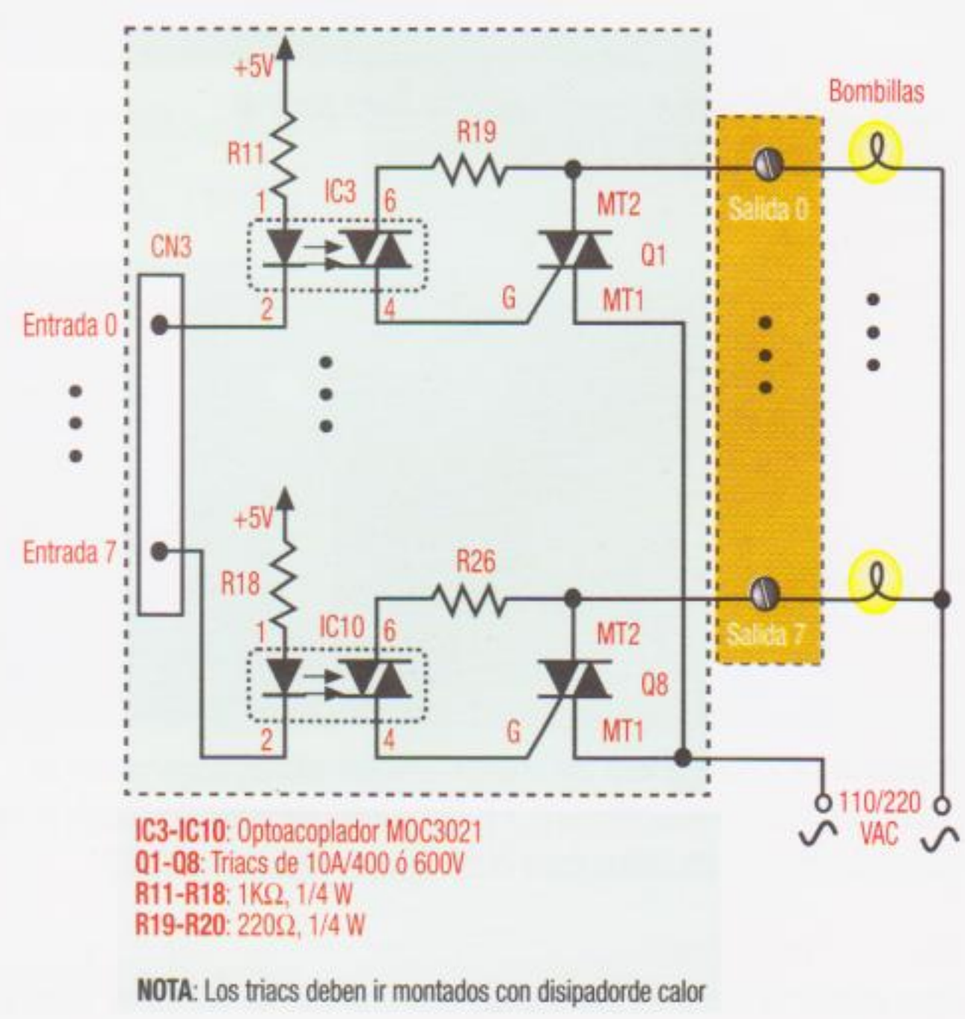


Figura 3.8 *Diagrama esquemático del byte de potencia*

Ensamblaje

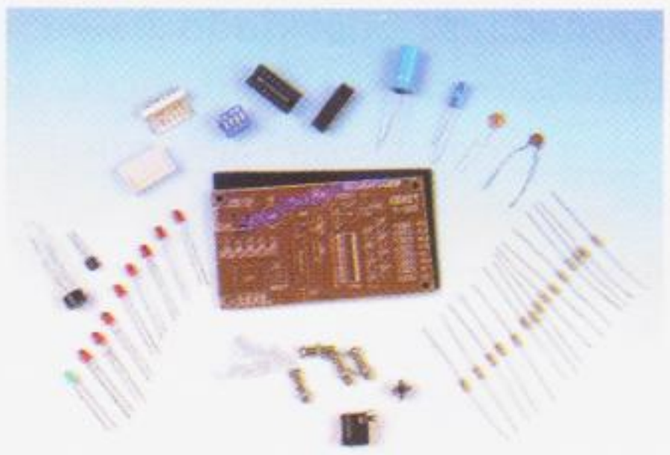


Figura 3.9 Este es el conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el proyecto.

LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Regulador fijo de +5V -100mA, 78L05
1	IC2	Microcontrolador PIC16F84 programado
1	BR1	Puente rectificador circular de 1A, W06
8	LED1 a LED8	Diodos LED rojos de 3mm
1	LED9	Diodo LED verde de 3mm
1	C1	Condensador electrolítico de 1000uF - 25V
1	C2	Condensador cerámico de 0.1uF - 50V
1	C3	Condensador electrolítico de 10uF - 16V
1	C4	Condensador cerámico de 20pF - 50V
9	R1y R8 a R15	Resistencias de 470 - 1/4W
6	R2,R3,R4,R5,R6 y R7	Resistencias de 10k - 1/4W
1	S1	Interruptor tipo DIP de 4 posiciones
1	S2	Pulsador pequeño de 4 pines, para circuito impreso
1	CON1	Conector CA/CC
1	CON2	Conector en línea de 8 pines
1	K-323	Circuito impreso CEKIT
1		Base para circuito integrado de 18 pines
1		Lámina de acrílico de 95mm X 55mm
4		Tornillos de 1/8" X 1/2" con sus tuercas y separadores

Paso 1

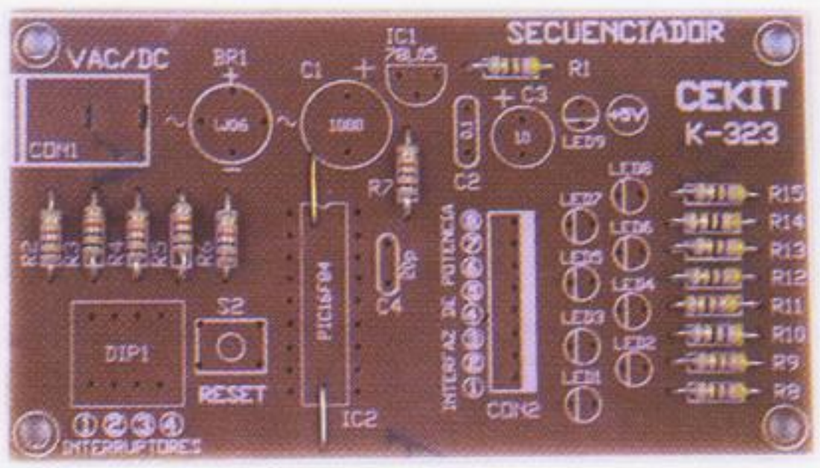


Figura 3.10 Ubique y suelde todas las resistencias (15) y los puentes de alambre

Paso 2

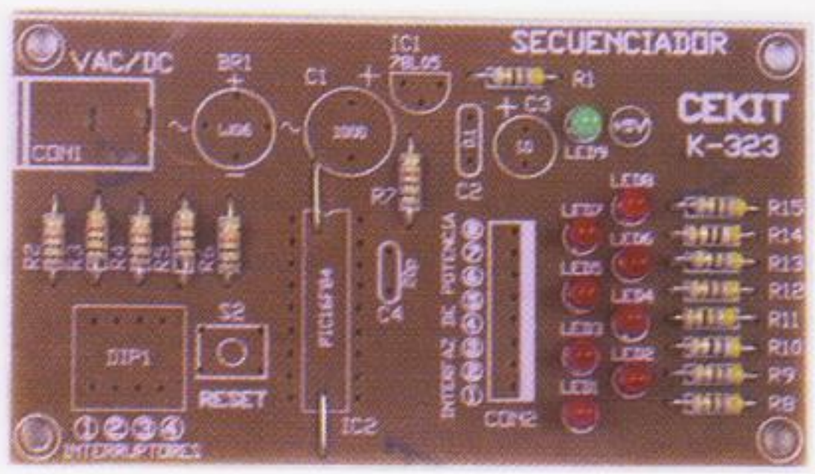


Figura 3.11 Suelde a continuación los diodos LED teniendo en cuenta su polaridad (ánodo y cátodo)

Paso 3

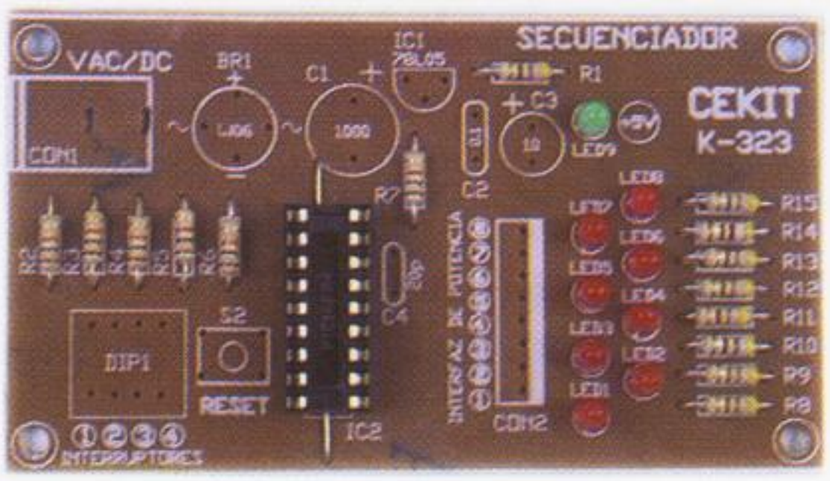


Figura 3.12 Instale y suelde ahora la base para el microcontrolador (IC2)

Paso 4

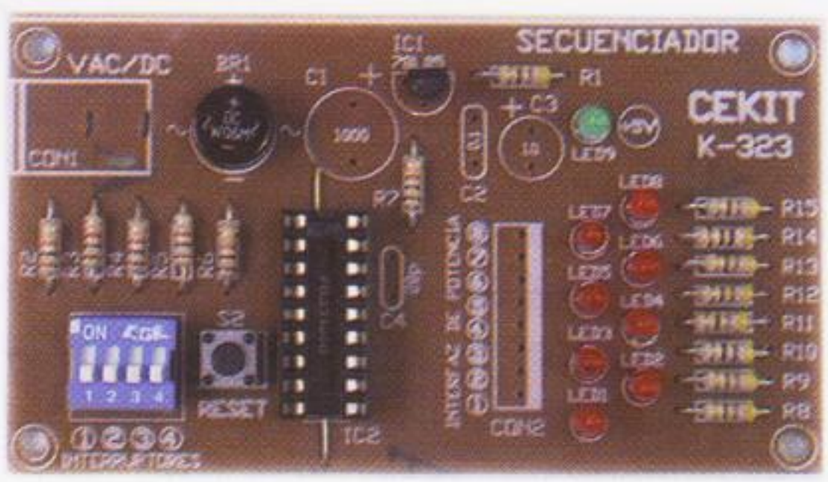


Figura 3.13 Instale y suelde los dos interruptores (dispswitch y el pulsador S2), el puente rectificador y el regulador de voltaje IC1.

Paso 5

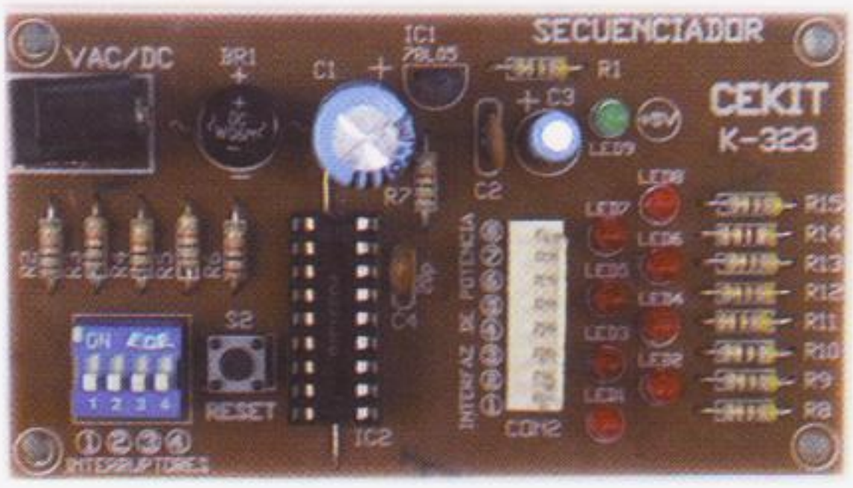


Figura 3.14 Instale ahora los cuatro condensadores observando la polaridad de los electrolíticos (C1 y C3), luego los conectores de alimentación y el de salida para el circuito de potencia (CON2)

Paso 6

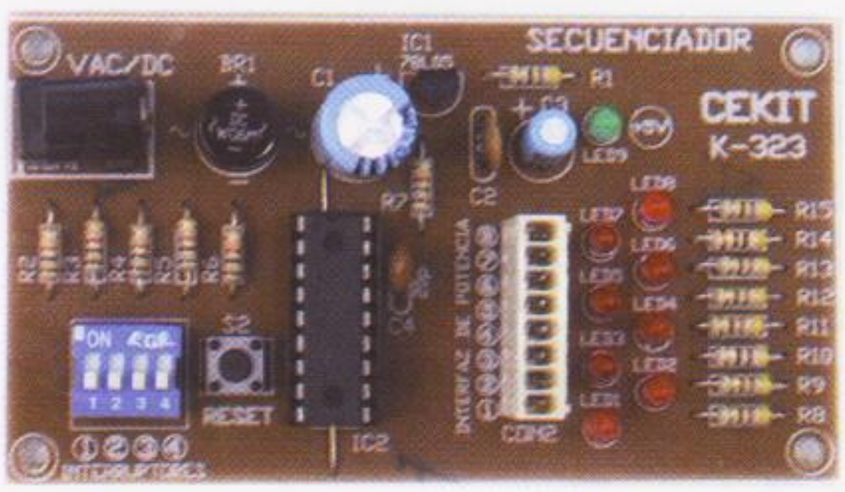
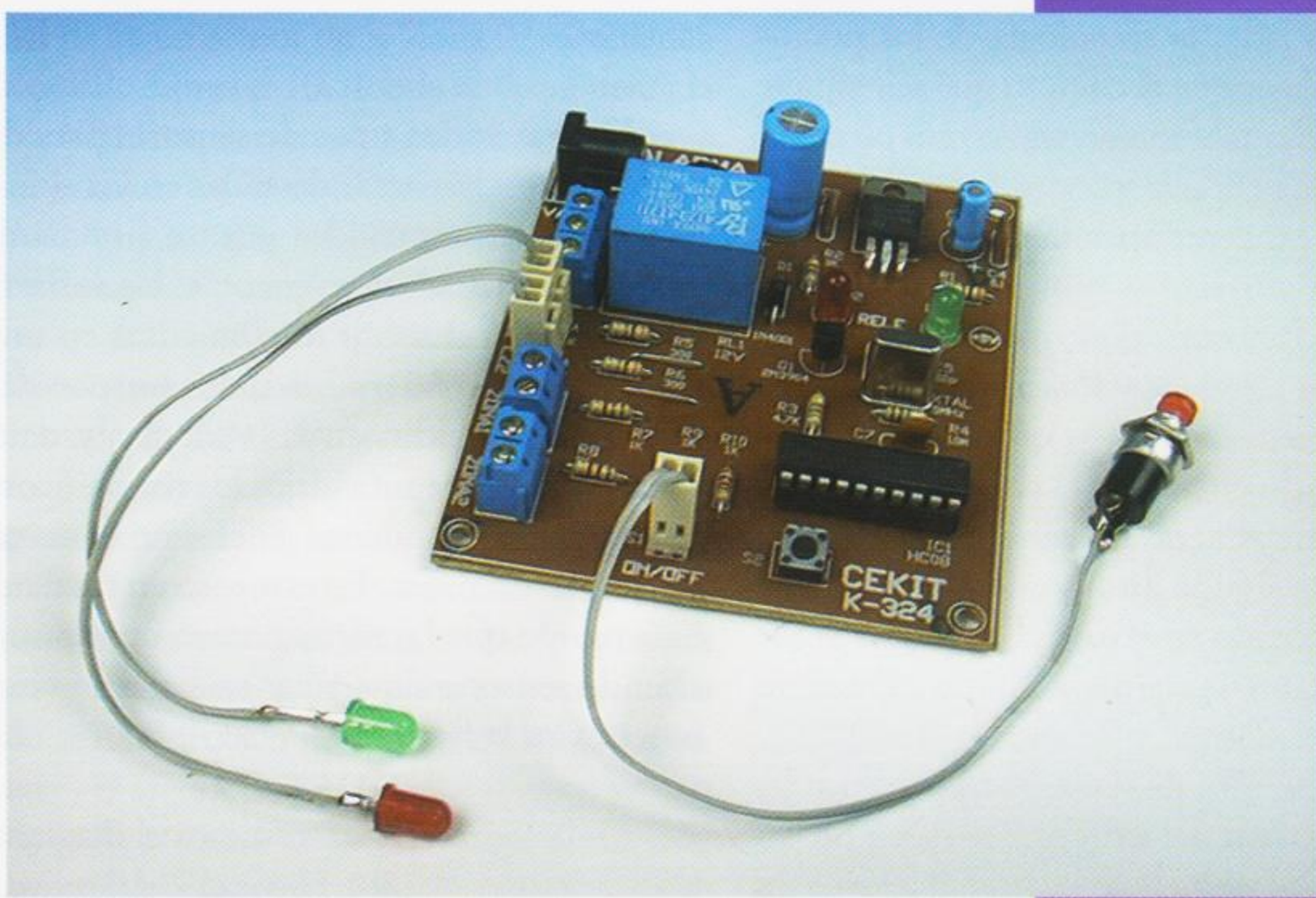


Figura 3.15 Instale en su base el microcontrolador PIC 16F84 programado, observando la posición correcta del pin No.1

Paso 7. Prueba final

Para probar el circuito, aliméntelo con una fuente regulada o no, hasta de 12 voltios y presione el pulsador RESET; luego siga las instrucciones de operación descritas anteriormente y el circuito debe mostrar la secuencia en los diodos LED; si no es así, revise muy bien las conexiones y verifique que el programa esté grabado en el microcontrolador.

PROYECTO 4



ALARMA PARA EL HOGAR

En este proyecto construiremos una alarma básica para el hogar utilizando un microcontrolador Motorola 68HC08. Es un dispositivo sencillo, si lo comparamos con las alarmas que ofrecen las empresas especializadas en seguridad, pero puede ser de gran utilidad en nuestro propio hogar u oficina.

Características

- Salida por medio de relevo
- Diodos LED indicadores del estado de la alarma
- Distintas zonas de operación

Funcionamiento

Uno de los problemas más grandes que afrontamos actualmente es la inseguridad y los sitios favoritos de los amigos de lo ajeno son las casas y oficinas que permanecen solas gran parte del día o de la noche. Afortunadamente, la detección electrónica de estas intromisiones no es muy complicada. Por otro lado, la tecnología de los microcontroladores hace que el circuito que se requiere para llevar a cabo tal función sea lo más pequeño posible, con una eficacia y seguridad que lo convierten en una excelente opción. El sistema de alarma de nuestro proyecto es muy sencillo y práctico, **figura 4.1**. Para entenderlo mejor, es necesario definir algunos términos y funciones:

Armado. Estado en el cual el sistema está vigilante y se puede generar una condición de alarma al detectar alguna anormalidad.
Desarmado. Estado en el que no se puede generar una condición de alarma ya que la unidad no se encuentra vigilante.
Zona. Es un circuito de sensores que se encarga de vigilar un sector del sitio que se desea proteger; se conecta al sistema de seguridad y le indica a éste si se detecta alguna violación.
Sensor. Es un dispositivo que detecta alguna condición particular de intromisión y le indica a la unidad central que debe generar una condición de alarma.

Este sistema consta de dos zonas:
Zona temporizada (Zona 1). En ella se deben conectar los sensores que vigilan el acceso al lugar que se protege; su función es detectar una intromisión y activar un estado de prealarma durante unos 20 segundos, tiempo durante el cual se debe desactivar el sistema o de lo contrario se iniciará una condición de alarma.
Zona inmediata (Zona 2). Una intrusión en esta zona genera una condición de alarma en forma instantánea.

Existen varios tipos de sensores para detectar las intromisiones indeseadas en los sitios que se encuentran protegidos por un sistema de seguridad. Entre ellos se cuentan los infrarrojos, que permiten detectar movimientos en un área de cubrimiento determinada; los sensores de ruptura, que se ubican en las ventanas y detectan cuando se rompe un vidrio; los magnéticos, que se instalan en las puertas y detectan el momento en que se abren, etc. En este caso en particular se ha asumido que las zonas poseen una configuración de sensores en serie y en paralelo como se muestra en la **figura 4.2**.

En la **figura 4.3** observamos el diagrama esquemático del circuito. El corazón del proyecto es un microcontrolador Motorola 68HC908JK3; los dos diodos LED indicadores se han conectado por los pines 10 y 12, que corresponden a los pines 6 del puerto D y 3 del puerto B; la salida del sistema está por el pin 9 que es el pin 7 del puerto D; finalmente los sensores de la alarma de la zona 1 se han conectado por el pin 2 del puerto B y los de la zona 2 por el pin 1 del puerto B.

El dispositivo que da la salida para indicar la condición de alarma es un relevo, el cual tiene

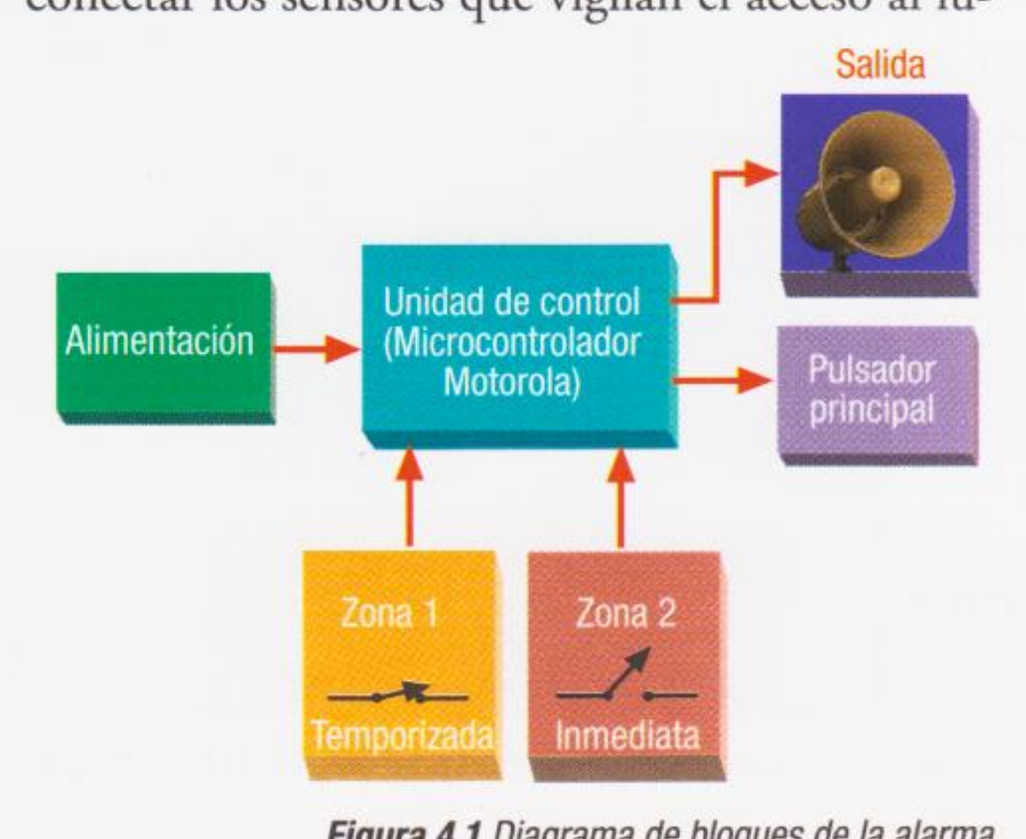


Figura 4.1 *Diagrama de bloques de la alarma*

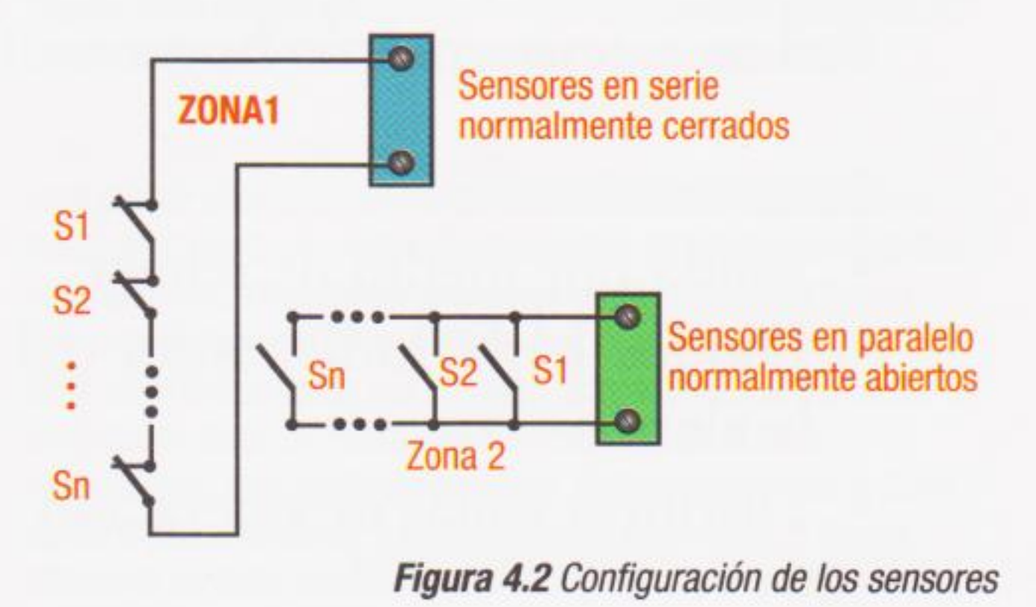


Figura 4.2 *Configuración de los sensores*

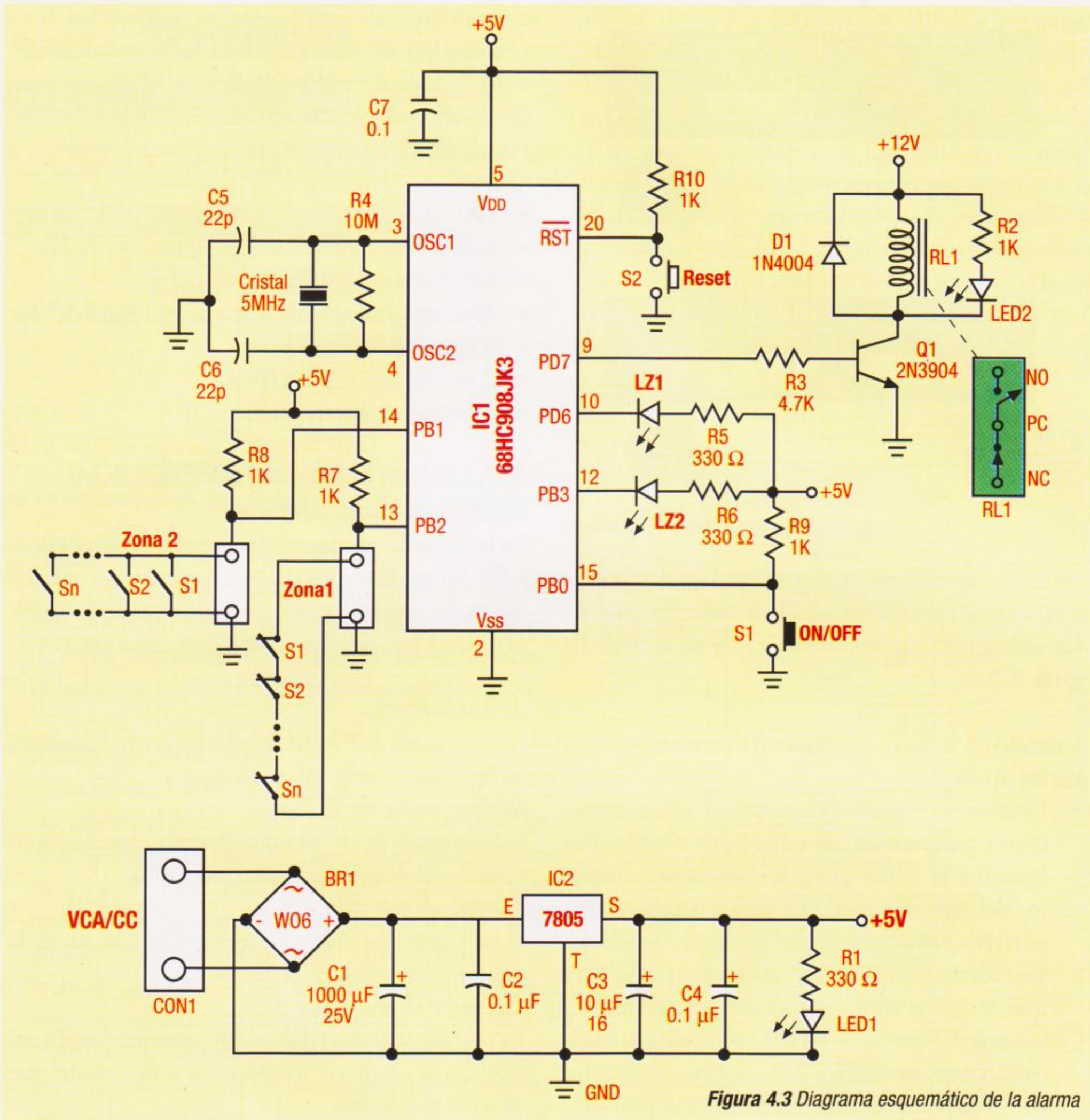


Figura 4.3 *Diagrama esquemático de la alarma*

sus contactos normalmente abiertos y se cierran en el momento que se debe señalizar. Los aparatos que se usan para indicar una alarma son generalmente sirenas, bombillos o discadores telefónicos. La salida del relevo es de libre utilización; a manera de ejemplo, en la **figura 4.4** se muestra como sería la conexión de una sirena de 12 voltios CC y de una sirena de corriente alterna.

Señalización del sistema

El sistema de alarma se controla con un pulsador (S1) el cual da la señal de armado/desarmado al microcontrolador. Cuando se pulsa S1, se activan los dos diodos LED (LZ1 y LZ2) indicando que se dispone de un corto tiempo para salir del lugar. Este tiempo se conoce como *tiempo de salida,* después del cual se entra a una verdadera condición de alarma. Uno de los sensores de la Zona 1 debe ser el de la puerta principal para poder entrar y salir antes de que se active la alarma. Cuando se ha terminado el tiempo de salida, se tiene el estado de *armado* de la alarma en el cual se detecta permanentemente el estado de los sensores de cada una de las zonas; cuando se

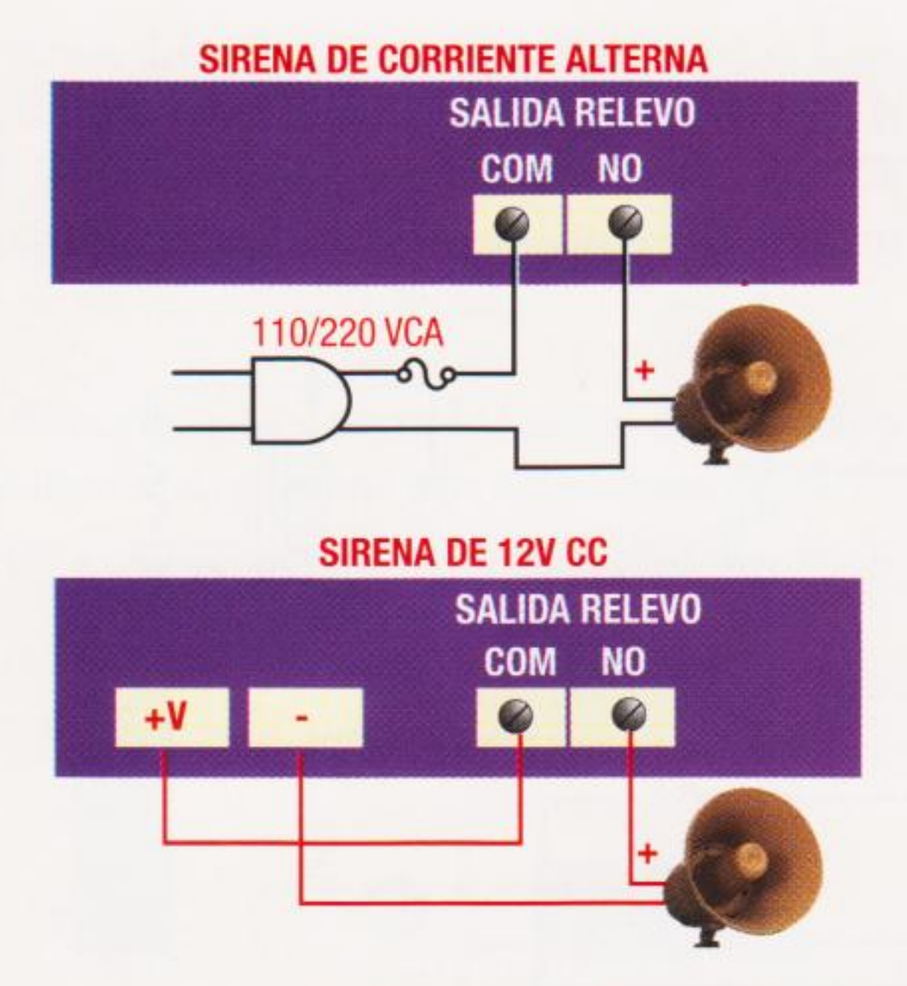


Figura 4.4 *Ejemplo de conexión de sirenas*

produce alguna anomalía, se activa el relevo y a través de los diodos LED se indica a cual de las zonas pertenece el sensor que se activó, **figura 4.5**.

Cuando se activa el estado de armado existen varias opciones:

- Una detección en alguna de las zonas inmediatas generará una condición de alarma. Por lo tanto, la salida que consiste en los contactos del relevo se cerrará y podrá activar el dispositivo señalizador que se haya instalado.
- Una detección en la zona temporizada hará que se inicie una condición de prealarma y los dos diodos LZ1 y LZ2 se activarán, indicando que se cuenta con algunos segundos para desarmar el sistema usando el pulsador S1. Si no se ejecuta esta operación, se generará una condición de alarma.

El microcontrolador

Antes de continuar con la explicación del funcionamiento de este proyecto, analicemos brevemente las características del microcontrolador utilizado. Se trata del 68HC908JK3 de Motorola, cuya distribución de pines se muestra en la **figura 4.6**. Sus principales características son:

- CPU de 8 *bits*
- 4 K*bytes* para memoria de programa (FLASH)
- 128 *bytes* de memoria RAM
- 10 canales para conversión de análogo a digital de 8 *bits*
- 15 pines de entrada/salida
- 2 temporizadores de 16 *bits*
- 8 fuentes de interrupción

Arquitectura interna del 68HC908JK3 y modelo de programación

En la **figura 4.7** se muestran los registros internos de la CPU 08 mediante los cuales se procesan las instrucciones, los operandos y los resultados. La función básica de cada uno de ellos se describe a continuación.

Acumulador (A)

Es un registro de 8 *bits* de propósito general usado en las operaciones aritméticas y lógicas.

Registro índice (H:X)

Es un registro de 16 *bits* utilizado como apuntador en el modo de direccionamiento indexado.

Puntero de pila (SP)

Es un registro de 16 *bits* que contiene la dirección de la posición disponible en la pila.

Contador de programa (PC)

Es un registro de 16 *bits* que contiene la dirección de la siguiente instrucción u operando que va a ser procesado.

DIODOS LED INDICADORES DE ZONA

LED 1	LED 2	ZONA
●	○	**ZONA 1**: Zona temporizada
○	●	**ZONA 2**: Zona inmediata

Figura 4.5 *Señalización de los diodos LED*

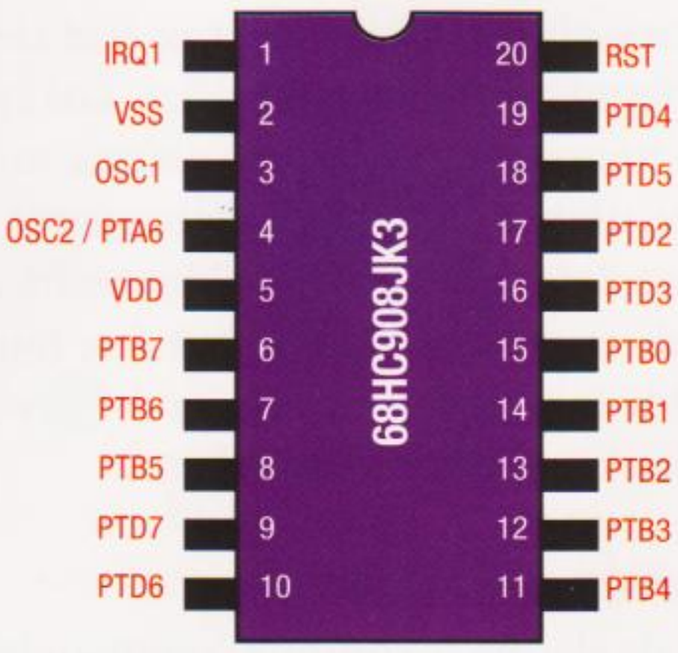


Figura 4.6 *Distribución de pines del microcontrolador 68HC908JK3*

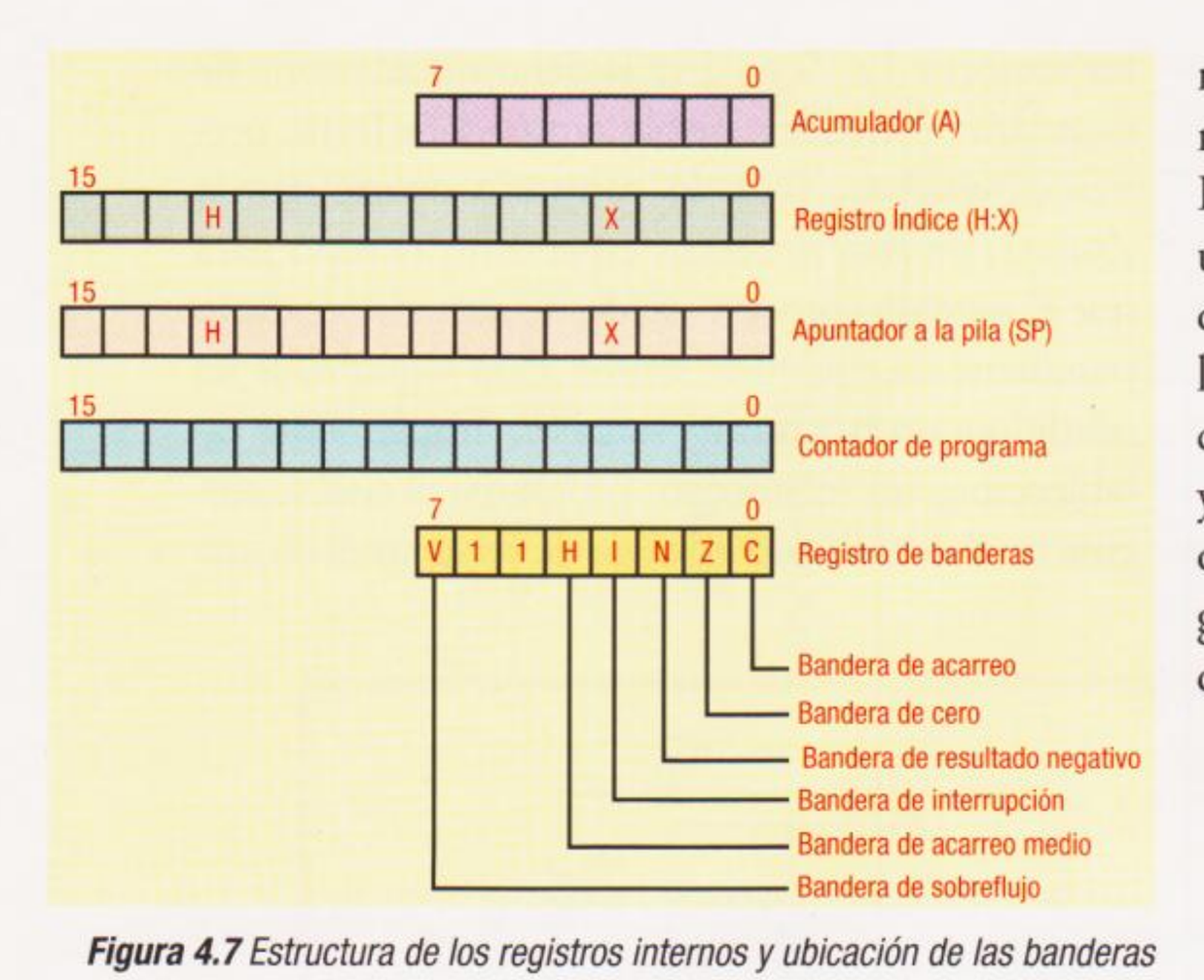


Figura 4.7 *Estructura de los registros internos y ubicación de las banderas*

Registro de banderas (CCR)

Es un registro de 8 *bits* que contiene el *bit* de enmascarado general de interrupciones y 5 banderas de estado, las cuales indican ciertas condiciones originadas por la instrucción previamente ejecutada.

Mapa de memoria

En la **figura 4.8** observamos el mapa de memoria completo del 68HC908JK3. Allí se encuentran las direcciones de los registros internos, la posición que ocupa la RAM y la parte asignada al programa de usuario. De igual manera, se observa la zona en la cual se encuentra almacenado el programa de fábrica (ROM MONITOR) que nos permite hacer simulación, depuración y programación del microcontrolador. En la zona alta de la memoria, se encuentran los vectores de interrupción y en ellos se encuentra la dirección (2 bytes) con la cual se carga el contador de programa (PC) cuando se presenta alguna de las interrupciones.

Descripción del programa

En la **figura 4.9** está el diagrama de flujo y en la **figura 4.10** una parte del código fuente. El programa completo lo puede obtener de nuestra página:

www.cekit.com.co

En el bloque **A** del programa establecemos las direcciones del inicio y final de las memorias RAM y ROM, de acuerdo a lo indicado en el mapa de la **figura 4.8**, al igual que definimos el mapa de las variables que vamos a utilizar. En el bloque **B** y numeral 2 del diagrama de flujo, configuramos los puertos D y B del microcontrolador según la función de cada uno de sus pines, de acuerdo a lo establecido en el diagrama esquemático de la figura 4.3. En el caso de los microcontroladores Motorola, se debe tener en cuenta que un pin de salida se configura con un "1" y un pin de entrada se configura con un "0", es decir, que se configura de manera inversa a como se configuran los microcontroladores PIC. En el bloque **C** del programa y numeral 3 del diagrama, se verifica la

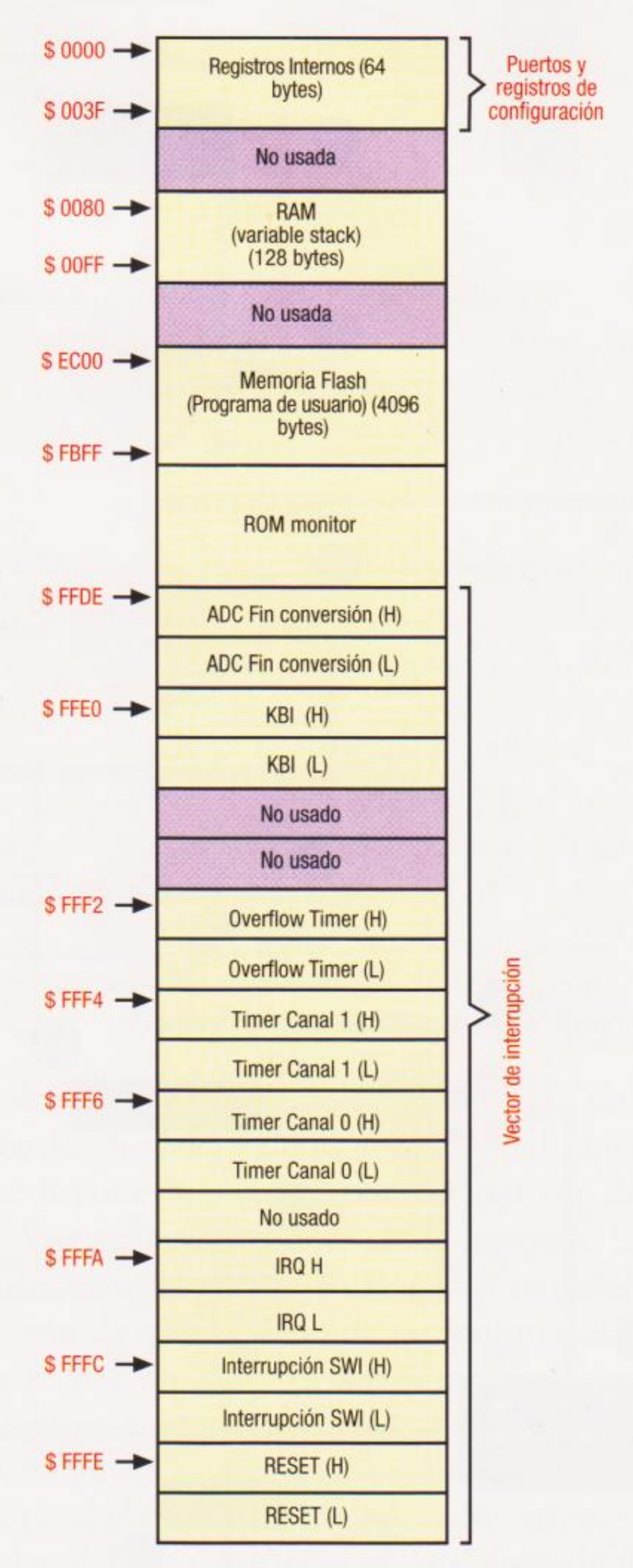


Figura 4.8 *Mapa de memoria interna del microcontrolador*

activación del pulsador de inicio (S1). Esto se hace con la instrucción **brclr PB0,PORTB,ZONA2**, la cual podemos interpretar como: chequee si existe un cero en el pin PD4 (definido como 4) del puerto D y de ser así, salte a ZONA2.

Cuando se ha detectado la señal de activación de la alarma, el programa salta a la etiqueta ZONA2, bloque F, numeral 4 donde se hace la detección de los sensores. La Zona 2 es la denominada zona de detección inmediata, por lo tanto, debe leerse permanentemente. Todo lo contrario ocurre con la Zona 1; en ésta debemos esperar un tiempo para que el sistema entre en estado de armado, es decir, pendiente de cualquier sensor. Para lograrlo, se ha acudido a un truco muy sencillo: simplemente establecemos un retardo por programa el cual se ejecuta un determinado número de veces antes de que

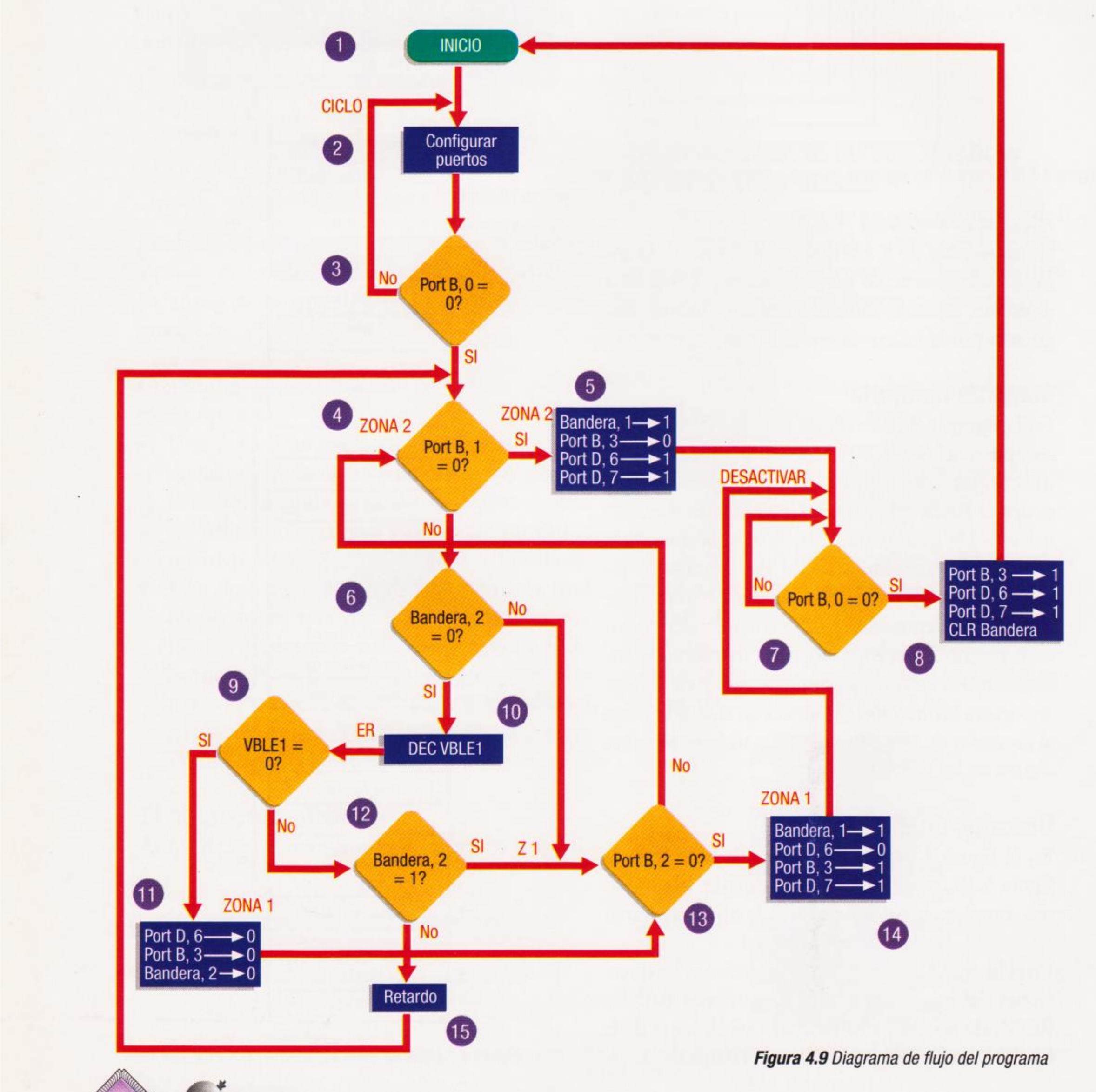


Figura 4.9 *Diagrama de flujo del programa*

se active la detección de la Zona 1. Cada vez que se llama la rutina RETARDO y ella se ejecuta, el programa salta a la Zona 2 para leer los sensores ubicados en ella y generar la señal de alarma si es necesario. Cuando se termina el tiempo de espera, el sistema queda completamente armado y pendiente de cualquier anomalía en el sitio vigilado. Si no detecta señal de algún sensor de la Zona 2, se salta a la etiqueta **ER**, numeral 10, donde se decrementa la variable VBLE1 quien contiene el número de veces que se ejecuta la rutina de retardo. Luego se llama a RETARDO y se salta a la etiqueta ZONA2 para hacer el chequeo correspondiente, bloques D y E.

En este programa hemos incluido un registro denominado BANDERA en el cual establecemos algunos *bits* como "testigos" de que se han presentado ciertos eventos. El *bit* cero de este registro se pone en alto en el momento que la alarma de la zona inmediata se ha activado, es decir, se ha detectado la activación de un sensor. El *bit* uno se pone en alto cuando se ha detectado un intruso en la zona temporizada y el *bit* dos de BANDERA se pone en alto cuando se ha terminado de invocar los tiempos de temporización de la zona 1. Por esta razón, en el bloque D, después de que se ha terminado de decrementar el registro VBLE1, se activa este *bit*.

Cuando termina de invocarse la rutina de retardo, las dos zonas quedan en estado de armado, es decir, en permanente vigilancia; si alguna se dispara, se enciende el LED correspondiente de acuerdo a la figura 4.5 y el programa se dirige a la rutina DESACTIVAR, bloque G, numeral 7 del diagrama de flujo. Allí se chequea de nuevo la existencia de una señal proveniente del pulsador S1; en este caso para apagar la alarma y borrar la señalización.

La rutina de retardo

Recordemos que un retardo por programa es una rutina que se encarga de ejecutar una serie de instrucciones (que no llevan a cabo ninguna función importante) encerradas dentro de un ciclo, un número determinado de veces. De esta manera se logra un espacio de tiempo que podemos utilizar dentro de otras rutinas.

```
INICIO:                                                    B
      clr    CONTA
      mov    #$C0,DDRD          ; Configuro puerto D
      mov    #$F8,DDRB          ; Configuro puerto B
      mov    #$9T,VBLE1         ; Asigno 9 a vble1

CICLO:                                                     C
      clr    BANDERA            ; Limpio el registro
                                ; bandera
      brclr  PB0,PORTB,ZONA2    ; Chequeo si está el
                                ; pulsador oprimido
      bra    CICLO

ZONA1:                                                     D
      brset  2,BANDERA,Z1       ; Si se ha cumplido el
                                ; tiempo de temporización,
      bclr   PD5,PORTD          ; salto a Z1
      bclr   PB3,PORTB          ; si no, chequeo la zona 2
      jsr    RETARDO            ; activo ambos led de
                                ; señalización
      bra    ZONA2              ; cuando estoy en el
                                ; período de temporiz.
ER    dbnz   VBLE1,ZONA1        ; decremento el número
                                ; de retardos
      bset   2,BANDERA          ; Activo este bit cuan-
                                ; do han terminado los
                                ; períodos de espera de
                                ; la zona 1

Z1    brclr  PB2,PORTB,zon1     ; Sistema armado
      bra    ZONA2              ; Salto a chequear la
                                ; zona 2
zon1  bset   0,BANDERA          ; Si se activa la
                                ; alarma de esta          E
      jsr    RETARDO            ; zona, el bit 0 de
                                ; bandera se pone
      jsr    RETARDO            ; en alto y se dan las
      bclr   PD6,PORTD          ; señalizaciones de los led
      bset   PB3,PORTB          ; activar led de zona 1
      bset   PD7,PORTD          ; se activa la señal de
                                ; alarma externa
      bra    DESACTIVAR         ; se salta al rutina de
                                ; desactivación

ZONA2:                                                     F
      brclr  PB1,PORTB,zon2     ; Se chequean los
                                ; sensores de esta zona
      brclr  2,BANDERA,ER       ; se verifica que no ha
                                ; terminado la temporiz.
      bra    Z1                 ; de la zona 1. Si ha
                                ; terminado, salta a Z1
zon2  bset   1,BANDERA          ; Cuando se detecta un
                                ; intruso en la zona 2
      bclr   PD5,PORTD          ; se activa el led de
                                ; la zona 2
      bset   PD6,PORTD          ; se desactiva el led
                                ; de la zona 1
      bset   PD7,PORTD          ; y se activa la señal
                                ; de alarma externa
      bra    DESACTIVAR         ; se salta a la rutina
                                ; de desactivación

DESACTIVAR:
      brclr  PB0,PORTB,ET1      ; se chequea de nuevo
                                ; el pulsador S1
      bra    DESACTIVAR         ; y si se oprime
ET1   bset   PB3,PORTB          ; se apagan los led       G
      bset   PD6,PORTD          ; se desactiva la
                                ; alarma
      bclr   PD7,PORTD          ; y se retorna a inicio
      clr    BANDERA            ; para ejecutar de
                                ; nuevo el programa
      bra    INICIO
```

Figura 4.10 *Código fuente del proyecto*

Ensamblaje

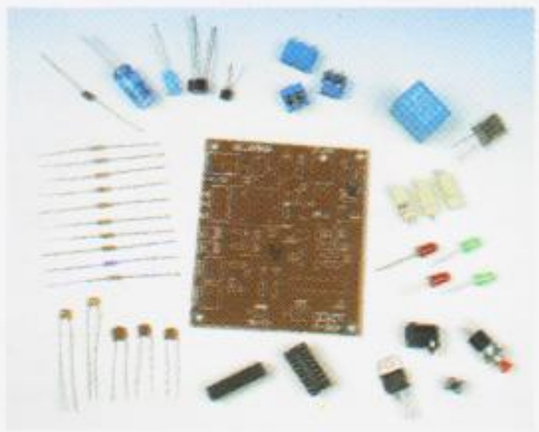


Figura 4.11 *Este es el conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el proyecto.*

Paso 1

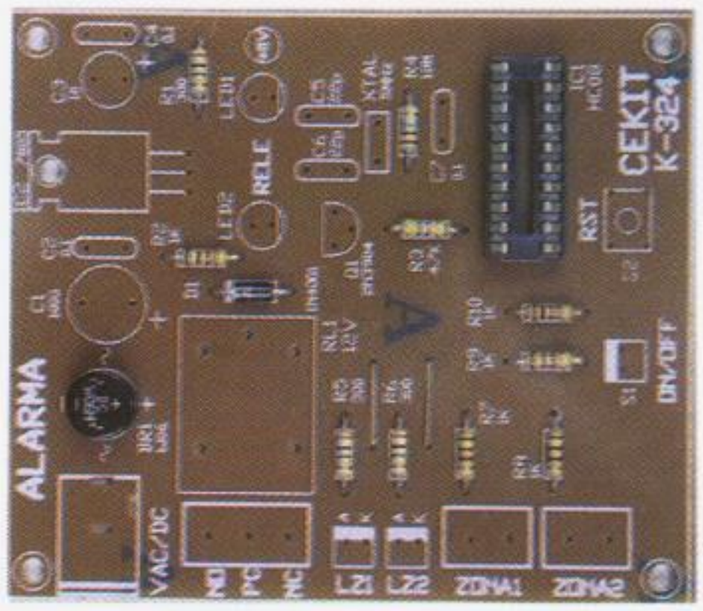


Figura 4.12 *Ubique y suelde los puentes de alambre, las resistencias (10), el diodo D1, el puente BR1 y la base para IC1*

Paso 2

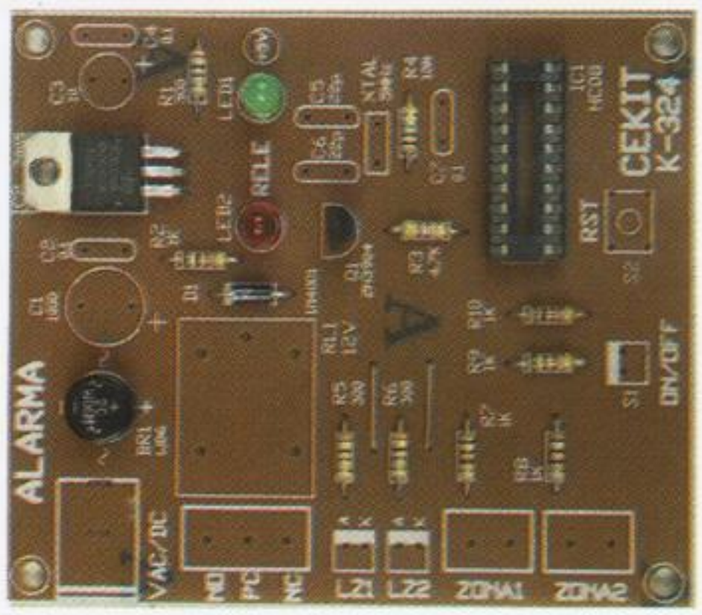


Figura 4.13 *Instale y suelde ahora el regulador IC2 (7805) y los diodos LED 1 y 2 observando su polaridad*

Paso 3

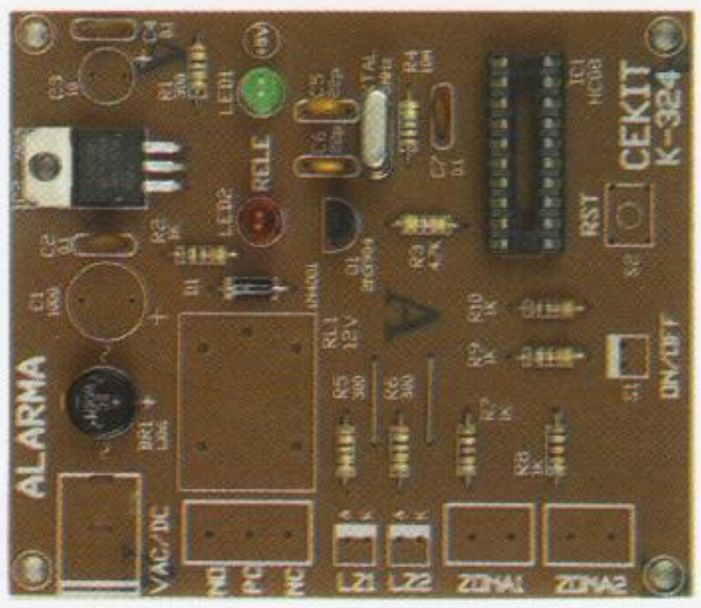


Figura 4.14 *Instale ahora los cinco condensadores cerámicos y el cristal de 5 MHz*

LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador 68HC908JK3
1	IC2	Regulador fijo de +5V -1A, 7805
1	Q1	Transistor de propósito general NPN, 2N3904
1	BR1	Puente rectificador circular de 1A, W06
1	D1	Diodo de propósito general 1N4004
2	LED1 y LED3	Diodos LED verdes de 5mm
2	LED2 y LED4	Diodos LED rojos de 5mm
1	C1	Condensador electrolítico de 1000uF - 25V
3	C2, C4 y C7	Condensadores cerámicos de 0.1uF - 50V
1	C3	Condensador electrolítico de 10uF - 16V
2	C5 y C6	Condensadores cerámicos de 22pF - 50V
1	XTAL	Cristal de 5MHz
3	R1, R5 y R6	Resistencias de 330Ω - 1/4W
5	R2, R7, R8, R9 y R10	Resistencias de 1k - 1/4W
1	R3	Resistencia de 4.7k - 1/4W
1	R4	Resistencia de 10M - 1/4W
1	S1	Pulsador normalmente abierto para chasis
1	S2	Pulsador pequeño de 4 pines, para circuito impreso
1	RL1	Relevo de 12V 5 A CC
1	K-324	Circuito impreso CEKIT
1		Conector CA/CC
3		Conectores en línea M/H de 2 pines
1		Conector de tornillo de 3 pines
2		Conectores de tornillo de 2 pines
1		Base para circuito integrado de 20 pines

Paso 4

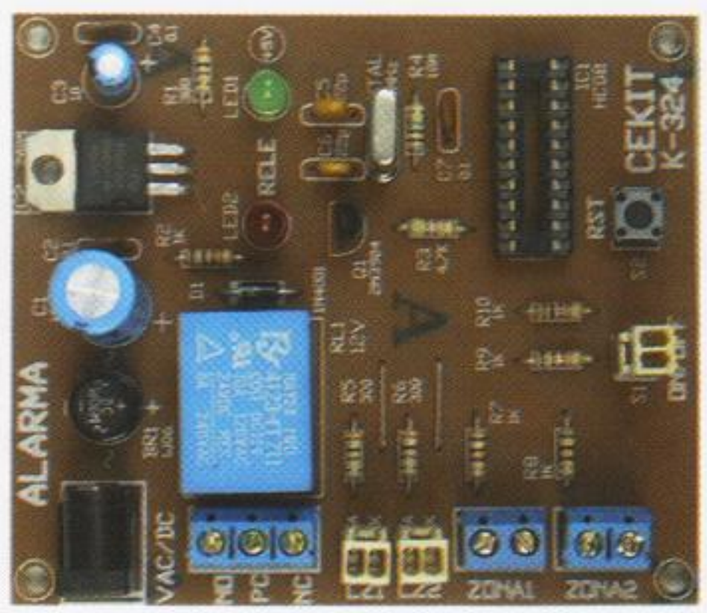


Figura 4.15 *Suelde a continuación el resto de los componentes teniendo en cuenta la polaridad de los condensadores C1 y C3*

Paso 5

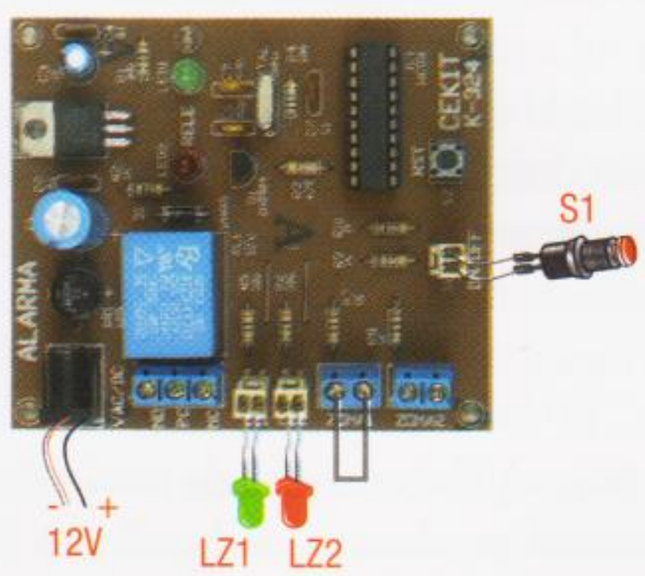
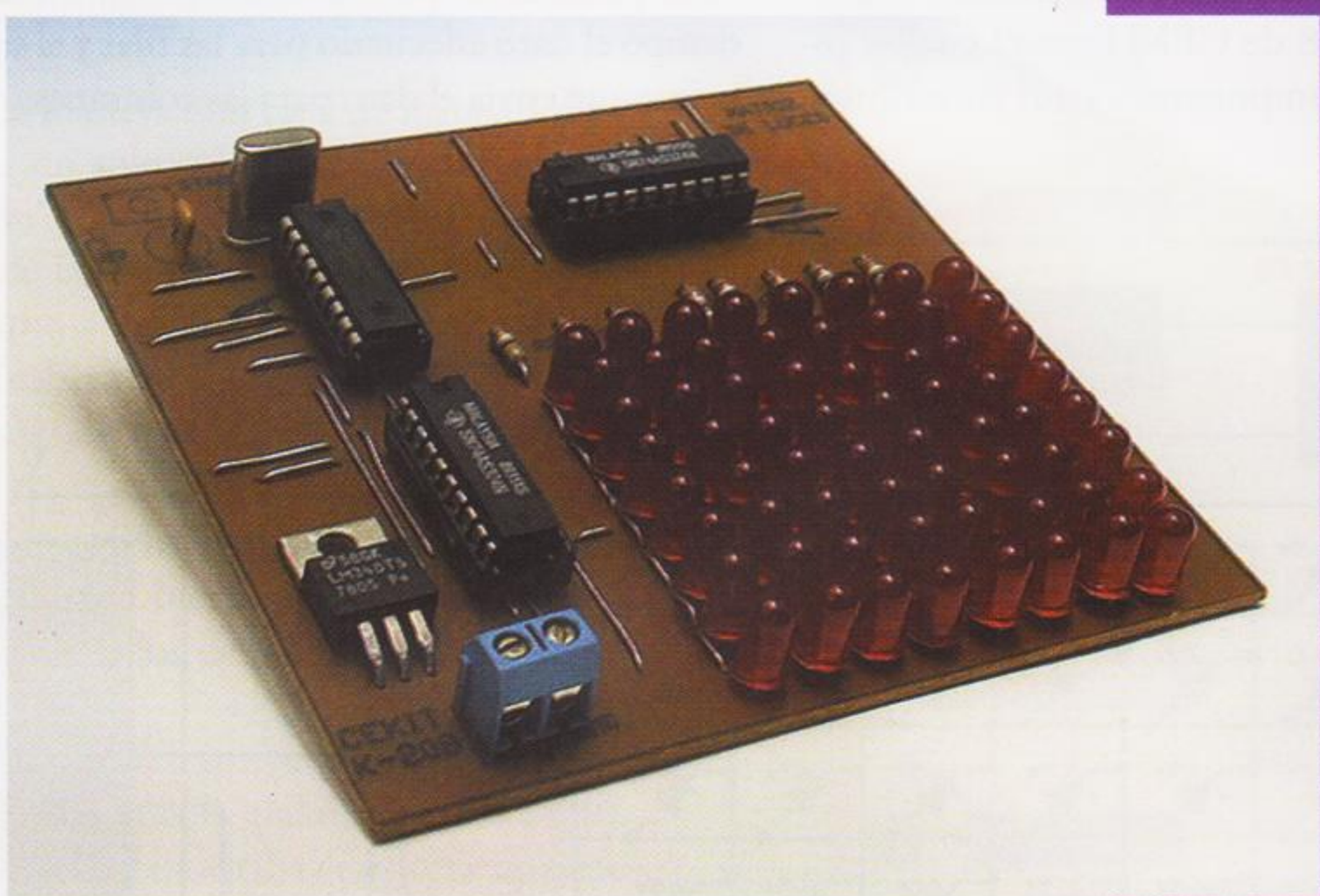


Figura 4.16 *Instale en su base el microcontrolador 68HC908JK3 programado, observando la posición correcta del pin No.1*

Paso 6. Prueba inicial

Para probar el circuito, aliméntelo con una fuente regulada o no, hasta de 12 voltios y siga las instrucciones de operación descritas anteriormente. Conecte un puente en los terminales de Zona 1. Presione el pulsador S1 y verifique que la señalización de los LED de la zona temporizada sea correcta. Mientras esta señalización se está dando, haga un puente en los contactos de la Zona 2 y se debe generar una salida por el relevo. Retire el puente de la Zona 2, repita el procedimiento, y retire el puente de la Zona 1. En ese momento se debe producir la activación del relevo y de la alarma.

PROYECTO 5



AVISO LUMINOSO CON DIODOS LED

Construya este proyecto y diviértase con él. Es una matriz de diodos LED de ocho columnas y ocho filas, donde se pueden mostrar textos o graficar algunas figuras que resultan muy simpáticas. Anímese y cree sus propios patrones.

Características

- Tablero de 8 x 8 diodos LED
- Tres efectos de animación diferentes dentro del tablero

Este proyecto es muy interesante para todos los interesados en la experimentación con microcontroladores. Se trata de un pequeño tablero animado, con diodos LED, de 8 x 8 pixeles en donde podemos conocer los principios básicos de este tipo de tableros, por supuesto más grandes, los cuales se encuentran en avenidas, estaciones de transporte, vitrinas de almacenes, etc.

La matriz se monta sobre un circuito impreso referencia K-208 de CEKIT, en el cual se incluyen todos los componentes y sus conexiones.

Funcionamiento

Existen muchas técnicas para la elaboración de este tipo de dispositivos. Generalmente se utilizan registros de desplazamiento en cascada para habilitar secuencialmente las columnas e ir enviando el dato adecuado a cada una de las filas por uno de los puertos. En este caso se ha optado por manejar todos los LED desde un solo puerto, **multiplexando** secuencialmente el control de las filas y las columnas. En otras palabras, el puerto envía en un instante de tiempo el dato adecuado para las filas y al siguiente instante envía el dato para las columnas.

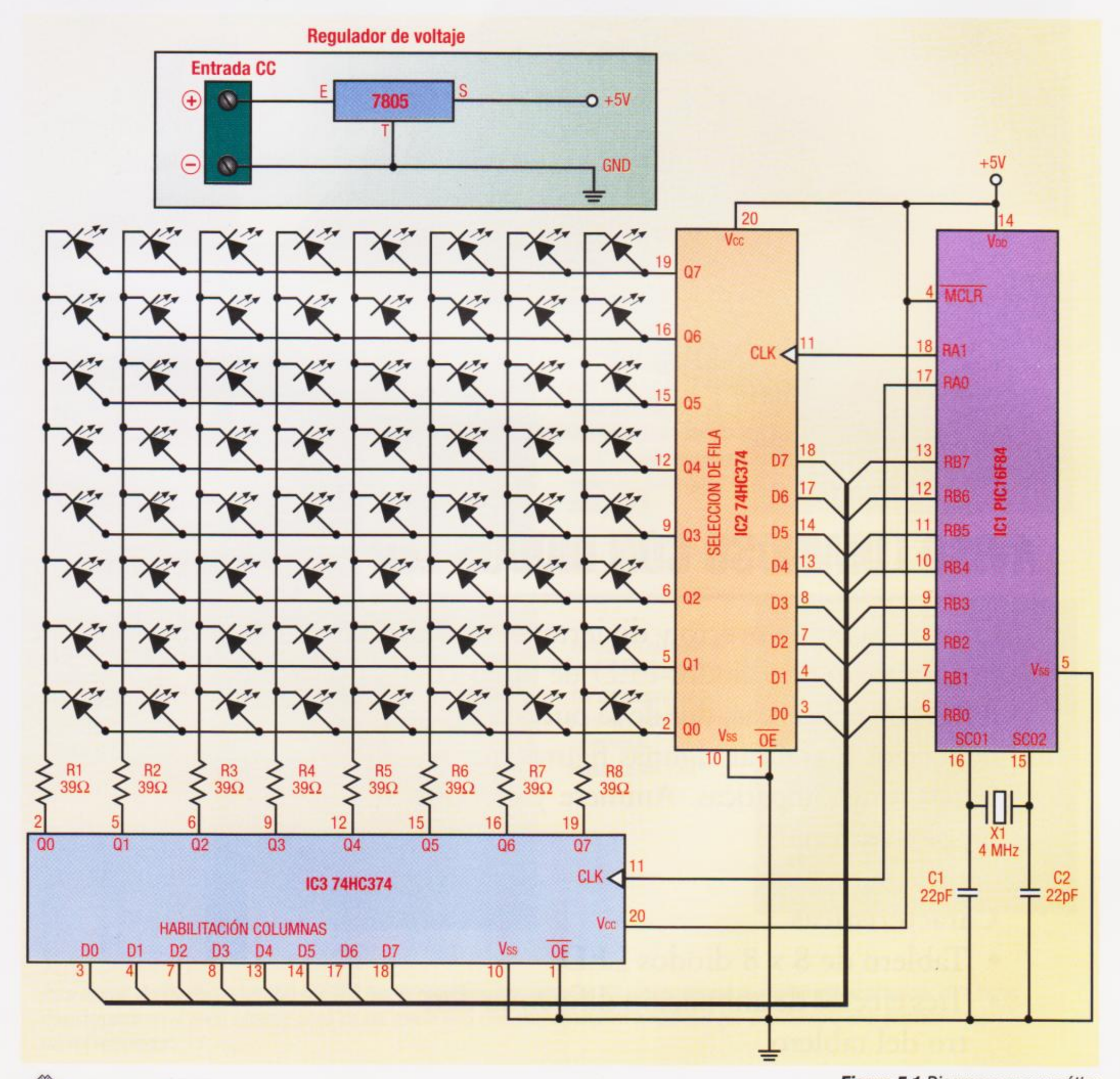


Figura 5.1 Diagrama esquemático

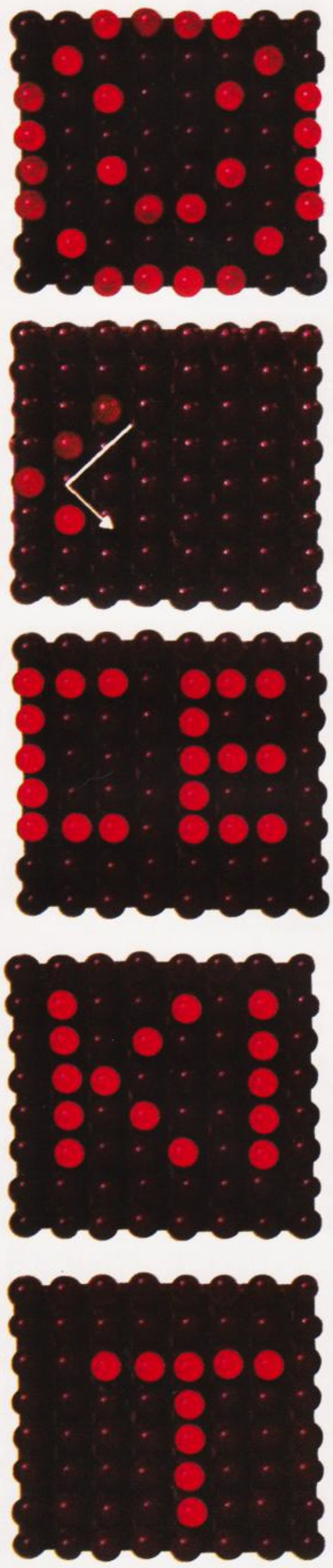


Figura 5.2 *Efectos gráficos creados dentro de la matriz*

El PIC16F84 no tiene suficientes líneas de I/O para manejar todos los LED directamente. El puerto B (líneas RB0 a RB7), se usa como **bus de datos** tal como se muestra en la diagrama esquemático de la **figura 5.1**. Este se conecta a las entradas D0 a D7 de dos *flip-flops* octales (74374). Las salidas de estos *flip-flops* manejan las líneas de las filas y las columnas. Las líneas o pines RA0 y RA1 se utilizan como salidas para las señales de reloj (generadas por programa) que sincronizan los datos en las líneas de filas y columnas.

Para configurar una fila o columna, al diseñar una figura, se deben enviar los datos deseados en la fila, (el *bit* de la fila deseada se debe poner en alto y las demás deben estar en bajo) en el puerto B, (líneas B0 a B7) y enviar un flanco de subida por la línea RA1 del puerto A para activar el reloj del *flip-flop* 74374.

Luego se envían los datos de la columna por el mismo puerto B (líneas B0 a B7) y se envía el pulso de activación del *flip flop* por la línea RA0 del puerto A. En este caso, cuando hay un 0 en una línea o *bit*, se enciende un LED y cuando hay un 1, está apagado. Antes de pasar a la siguiente fila, se debe enviar el dato FF (11111111) por las líneas de las columnas para limpiar el *display*; de otra manera se presentarían problemas al pasar de una fila a la siguiente.

Debe tener cuidado en el programa para no seleccionar varias filas simultáneamente; de lo contrario, se sobrecargará la columna, y por lo tanto, la capacidad del circuito integrado de suministrar corriente (*fan out*) será excedida, ocasionando que algunos o todos los LED disminuyan el nivel de su iluminación. Por esto se deben configurar cuidadosamente las frecuencias de aparición y rotación de los dibujos.

Al trabajar con un *display* de 8 x 8, inevitablemente salen a relucir limitaciones de memoria en el PIC 16F84. Por eso, para un proyecto similar con más capacidad, sería una buena práctica,diseñar un aviso con un microcontrolador de más memoria.

Otro asunto que debe tener en cuenta es que el *flip-flop* que mejor se ajusta en cuanto a capacidades eléctricas de consumo y suministro de potencia, características de tiempos, y rango aceptable de alimentación, es el 74HC374 de Motorola. Sin embargo, si no tiene este dispositivo, puede utilizar cualquier otro circuito integrado de la misma familia 74XX374.

Efectos gráficos

Se han incluido dentro de este tablero algunos efectos gráficos sencillos; sin embargo, se pueden crear muchos más efectos si utiliza un microcontrolador con una memoria de programa de mayor capacidad.

El primer efecto es la palabra CEKIT que aparece en la pantalla y que rota secuencialmente hacia arriba; después de éste, aparece el dibujo parpadeante de una cara feliz y finalmente, una esfera que rebota dentro del tablero. Esta secuencia se repite indefinidamente, **figura 5.2**.

Descripción del programa

Debido a la extensión del programa de este proyecto, solo publicamos sus partes principales, pero el código completo se encuentra disponible en nuestra página de internet:

www.cekit.com.co

En la **figura 5.3** se muestra el diagrama de flujo del proyecto y en la **figura 5.4** parte del código fuente.

El programa se ha dividido en una serie de subrutinas, cada una de las cuales cumple con una función específica y luego retorna al punto desde donde fue invocada. Esas rutinas se encargan de cargar los datos que conforman el dibujo, de desplegarlo en la pantalla y de rotarlo.

En los bloques A, B y C del programa, se hacen las asignaciones correspondientes a las direcciones en la memoria RAM para cada una de las variables. En el bloque D se hace un llamado a la subrutina **ShowVersion**, la cual despliega un letrero inicial que no se repite más adelante y que sirve para mostrar la versión de la tarjeta; es decir, esta sería la versión 1. ShowVersion está reseñado con la letra I; allí se cargan los datos del letrero de versión en ocho posiciones de memoria RAM, empezando desde la 10H, donde se encuentra el registro Screen0. Luego se invoca la subrutina **RepitaDibujo** en la cual se llama a la rutina DisplayDibujo, bloque G, la cual hace aparecer en el tablero los datos cargados durante un tiempo determinado.

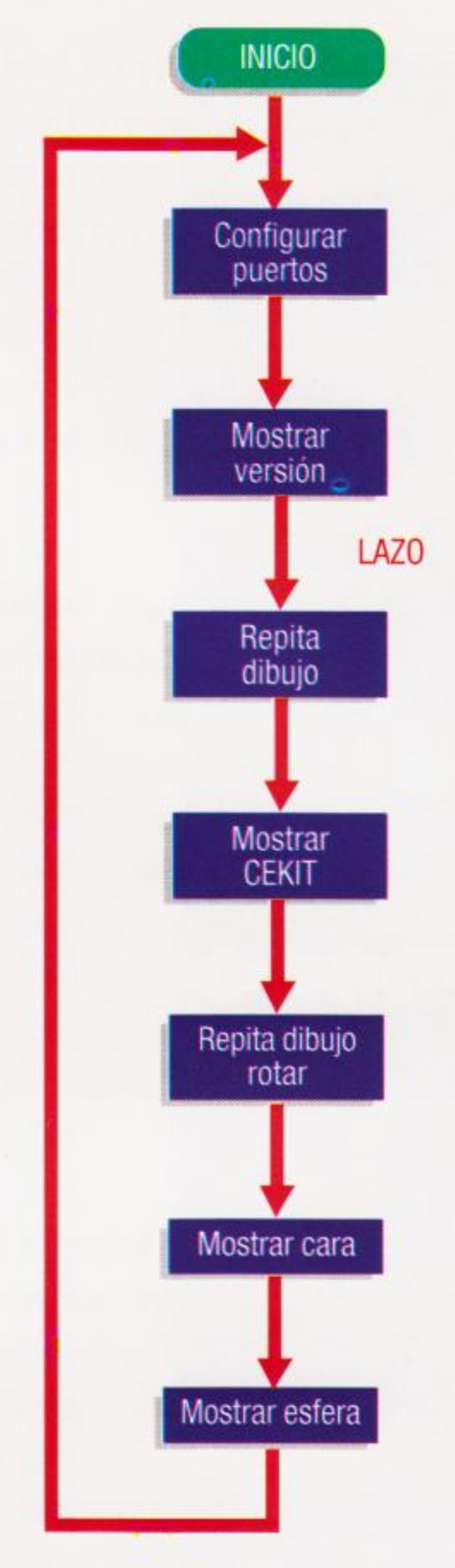


Figura 5.3 *Diagrama de flujo de la matriz*

PARA SABER MÁS

El modo de **direccionamiento indirecto** de la memoria de datos se utiliza cuando las direcciones en esa memoria no son fijas. Un registro especial se utiliza como apuntador a las posiciones de la memoria de datos que van a ser leídas o escritas. Como este apuntador está en la RAM, sus contenidos pueden modificarse por programa. Esto es muy útil para el manejo de tablas de datos.

El funcionamiento en realidad es muy sencillo: el registro especial de que hablábamos hace un momento se denomina FSR (***F**ile **S**elect **R**egister*: Registro de selección de archivo). El valor que especifica ese registro es una dirección que contiene un dato y ese dato es almacenado automáticamente en otro registro denominado INDF. En otras palabras, cualquier instrucción que utilice el registro INDF, tiene acceso al registro apuntado por FSR.

```
            ;-----------------------------
            ORG     0
PowerOn     BSF     STATUS,RP0
            MOVLW   H'00'
            MOVWF   TRISB   ;Configura todos los bits
                            ;del PORTB como salidas
            MOVWF   TRISA   ;Configura todos los bits
                            ;del PORTA como salidas
            ;-----------------------------
            CALL    ShowVersion ; La primera visua-
                                ;lización que se tiene
                            ;es la versión del proyecto.

            MOVLW   H'10'   ;Luego se hace visualzar
                            ;ese patron 10 veces
            CALL    RepitaDibujo

            ;-----------------------------
LAZO
            BCF     STATUS,RP0 ;Se establecen las velo-
            MOVLW   RATA_DE_MULTIPLEXACION ;cidades de
            MOVWF   MultiplexRata ; multiplexación de
            MOVLW   RATA_DE_DIBUJO ; los dibujos
            MOVWF   RataDibu

            ;-----------------------------

            CALL    cekitLogo ; el primer efecto grá-
            MOVLW   H'10'     ; fico es el de el nom-
            CALL    RepitaDibujo ; bre CEKIT rotando
            MOVLW   H'40'     ; desde abajo hacia
            CALL    RepitaRotUp ; arriba

            ;-----------------------------

            CALL    ShowCarita ;se llama una rutina
            MOVLW   H'10'     ; donde se asignan los
            CALL    RepitaDibujo ; valores binarios
            CALL    BlankScreen ; que saldrán por el
            MOVLW   H'04'     ; puerto y permitirán
            CALL    RepitaDibujo ; visualizar la cari-
            CALL    ShowCarita ; ta. Luego se borra la
            MOVLW   H'10'     ; pantalla y vuelve a
            CALL    RepitaDibujo ; aparecer

            ;-----------------------------
            MOVLW   RATA_DE_DIBUJO / 2 ; ahora se
            MOVWF   RataDibu ; cambia el dibujo por

            MOVLW   H'00' ; una bola rebotando en el
            CALL    RepitaBola ; interior de la panta-
                               ; lla a una velocidad
            MOVLW   RATA_DE_DIBUJO ; determinada
            MOVWF   RataDibu
            ;-----------------------------

            GOTO    LAZO      ; toda la secuencia se
                              ; repite indefinidamente

;-----------------------------------
;       SUBRUTINA     MUESTRA EL MISMO DIBUJO [W]
;-----------------------------------
RepitaDibujo MOVWF  Scratch0 ; se cargan los valores
Lazo1        CALL   DisplayDibujo ; del efecto
             DECFSZ Scratch0,1 ; correspondiente
             GOTO   Lazo1      ; y se invoca la rutina
                               ; de visualización
             RETLW  0
;-----------------------------------
; muestra el dibujo por un intervalo corto. En esta
;subrutina se hace un barrido de los valores cargados en
;las posiciones donde se han configurado los distintos
;patrones gráficos. Esto se hace direcionando con el
;registros FSR y tomando el dato del registro INDF.
;-----------------------------------
DisplayDibujo MOVF   RataDibu,0 ; se carga el número
              MOVWF  RetDibujo  ; de veces que se hace
                                ; el barrido de los da-
                                ; tos de los dibujos

DFOuterLoop   MOVLW  ALTURA_PANTALLA
              MOVWF  ContaDibu  ; Número de posiciones
                                ; que se deben barrer
```

```
              MOVLW  H'01'       ; para visualizarse en
              MOVWF  TempDibu    ; la pantalla.

              MOVLW  Screen0 + ALTURA_PANTALLA ;FSR se
              MOVWF  FSR         ; carga con el valor más
                                 ; alto de la tabla.

DFLoop        MOVLW  B'00000000'; Inicialmente se desha-
              MOVWF  PORTA       ; bilitan todas las filas.
              DECF   FSR,1       ; Empieza el barrido
              MOVF   INDF,0      ; de arriba hacia abajo.
              MOVWF  PORTB       ; Se toma el dato y se
                                 ; lleva al puerto B.

              MOVLW  B'00000001'; Luego se habilita la
              MOVWF  PORTA       ; primera columna.
              MOVF   TempDibu,0 ; El orden es, primero
              MOVWF  PORTB       ; el dato en el puerto B
              MOVLW  B'00000010'; y luego la habilitación
              MOVWF  PORTA       ; de la columna en el
                                 ; puerto A.
              RLF    TempDibu,1 ; Se rota hacia la
                                 ; derecha el habili-
              BCF    TempDibu,0 ; tador de columnas.
              CALL   ShortDelay ; Se espera un momento.
              MOVLW  B'11111111'; Se activa todo el
              MOVWF  PORTB       ; puerto B por un
              MOVLW  B'00000001'; momento y se desactiva
              MOVWF  PORTA       ; para evitar el efecto
              MOVLW  B'00000000'; de fantasmeo de algunos
              MOVWF  PORTB       ; LED. Es decir que no
              MOVLW  B'00000010'; queden medianamente
              MOVWF  PORTA       ; activados.
              DECFSZ ContaDibu,1; Se decrementa el
                                 ; contador de barrido.
              GOTO   DFLoop      ; si no llega a cero,
                                 ; se direcciona la
                                 ; siguiente posición.

              DECFSZ RetDibujo,1 ; se decrementa el
                                 ; número de veces que
                                 ; se debe hacer todo el
              GOTO   DFOuterLoop ; barrido. Y se repite
              RETLW  0           ; mientras este no
                                 ; llegue a cero.

              ;-----------------------------------
;       patrón binario que configura la carita feliz
              ;-----------------------------------
ShowCarita
              MOVLW  B'11000011' ; a partir de la
              MOVWF  Screen0     ; dirección del
              MOVWF  Screen0+H'7'; registro Screen0
              MOVLW  B'10111101' ; se colocan
              MOVWF  Screen0+H'1'; consecutivamente
              MOVWF  Screen0+H'6'; los bytes
              MOVLW  B'01011010' ; que saldrán
              MOVWF  Screen0+H'2'; por el puerto
              MOVLW  B'01111110' ; y que configurarán
              MOVWF  Screen0+H'3'; en la matriz de LED
              MOVLW  B'01011010' ; la gráfica de la
              MOVWF  Screen0+H'4'; carita feliz
              MOVLW  B'01100110' ; con todos estos valo-
              MOVWF  Screen0+H'5'; res cargados se
              RETLW  0           ; retorna de esta
                                 ; subrutina
      ;-----------------------------------
;       patrón binario de visualiza la versión
      ;-----------------------------------
ShowVersion   MOVLW  B'01011111' ; esta es otra rutina
              MOVWF  Screen0     ; donde se cargan las
              MOVWF  Screen0+H'1'; posiciones a partir
              MOVLW  B'10111111' ; de Screen0 con los
              MOVWF  Screen0+H'2'; datos que mostrarán
              MOVLW  B'11111111' ; la versión del
              MOVWF  Screen0+H'3'; proyecto.
              MOVLW  B'01111011' ; Esta gráfica solo se
              MOVWF  Screen0+H'4'; muestra al inicio
              MOVLW  B'01110101' ; del sistema y luego
              MOVWF  Screen0+H'5'; no se vuelve a repe-
              MOVWF  Screen0+H'6'; tir.
              MOVLW  B'01011011'
              MOVWF  Screen0+H'7'
              RETLW  0
```

Figura 5.4 Código fuente

La rutina DisplayDibujo

Cuando se invoca esta rutina desde cualquier parte del programa, debe cargar el registro W con el número de datos que van a salir en cada efecto gráfico. Ese valor se almacena en el registro *RataDibu* que es el que determina el número de veces que se repite la rutina de visualización.

En el registro *ContaDibujo,* se guarda el número de filas que tiene la matriz; por lo tanto, en cada visualización, no se puede exceder este valor. En la **figura 5.5** se aprecia un diagrama del funcionamiento de esta rutina, y en la **figura 5.6** el diagrama de flujo de la misma.

La rutina empieza con el registro FSR direccionando la primera posición a partir de la cual se ha almacenado la figura que se pretende desplegar en el tablero, numeral 2; esta es la 10H que corresponde al registro Screen0.

El contenido de esta dirección es un dato (D8) que está disponible en INDF y que se envía por el puerto B para que sea el dato de las filas, numerales 5 y 6; inmediatamente se envía al puerto A el número 01H para activar el bus de las filas y luego se pasa al registro *TempDibu*, el cual contiene el número para activar cada una de las columnas en el puerto B.

Luego se envía el número 02H al puerto A para habilitar la entrada de ese dato en las columnas, bloque 9. El resultado es la visualización del primer dato en la primera columna el cual permanece durante un corto tiempo, numeral 11.

En el numeral 10 se rota hacia la izquierda el registro *TempDibu* para tener listo el valor que activará la siguiente columna. Esto se hace decrementando el registro *ContaDibujo*, numeral 17.

Mientras su valor no llegue a cero, el programa traerá el siguiente dato de la memoria a través del registro apuntador FSR, el cual se decrementó previamente en el numeral 4.

El proceso se repite, pero esta vez con el segundo dato (D7) desplegado en la columna 2, y continúa hasta completar el recorrido de los ocho datos.

Cuando el registro *ContaDibujo* ha llegado a cero, significa que se han mostrado todos los valores del gráfico que se quiere visualizar. En este momento, el flujo del programa lo lleva al numeral 15, en donde se decrementa el registro *RetDibu* el cual indica cuantas veces se va a repetir todo el proceso completo de visualizar todas las columnas de manera consecutiva. Cuando el proceso se ha repetido ese número de veces, se sale de la subrutina.

En el bloque E del programa se invocan cada una de las figuras y en cada una de ellas se hace alusión a la rutina anterior, pero cargando distintos datos.

Recomendaciones para el montaje

En las **figuras 5.7 a 5.13** puede observar la guía de ensamblaje de la tarjeta, los LED de

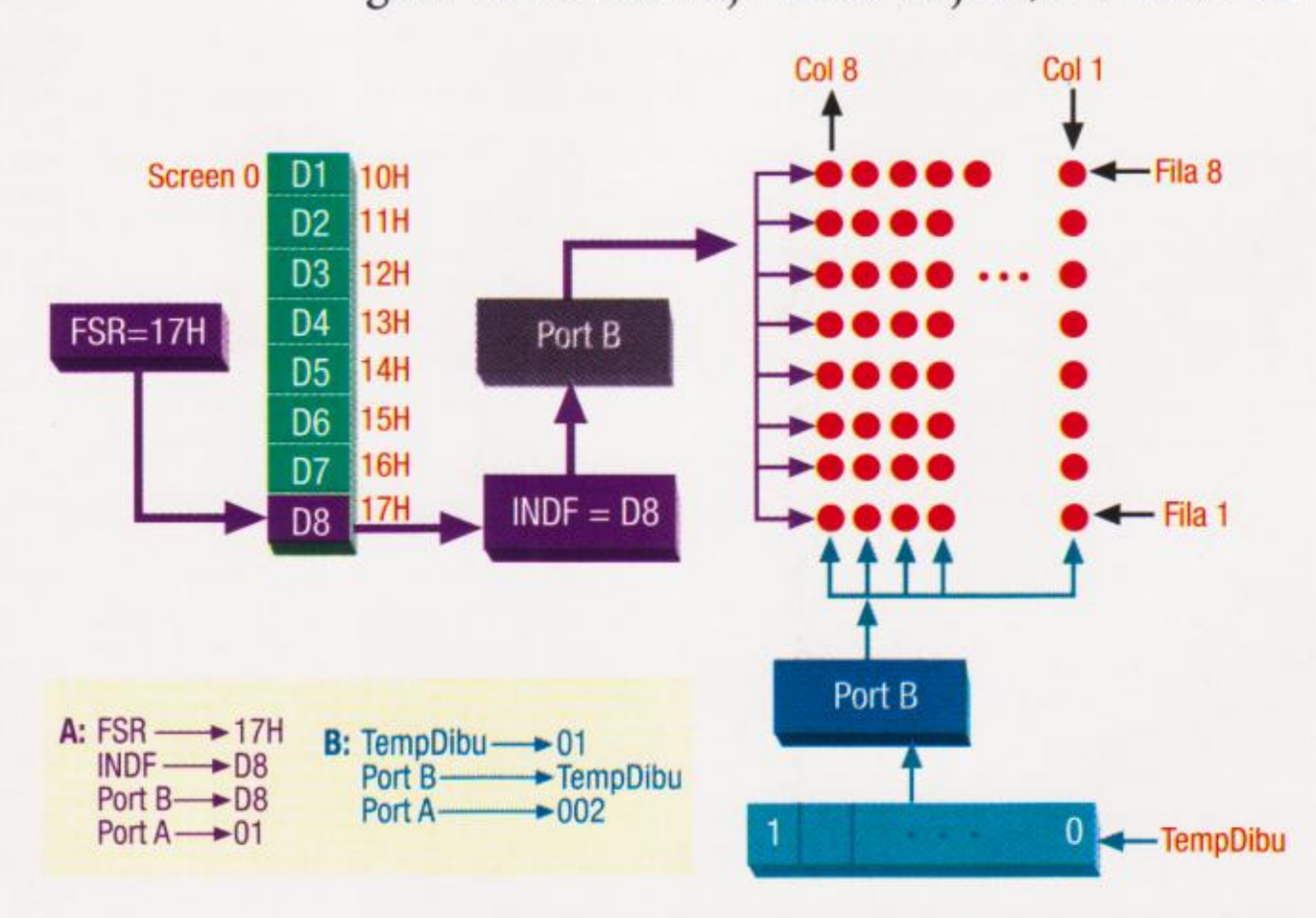


Figura 5.5 *Funcionamiento de la rutina DisplayDibujo*

5mm se han alineado y colocado de la manera más cercana posible, al igual que los circuitos integrados, con el objeto de darle una apariencia más compacta.

De todas maneras se pueden utilizar los LED de 3mm, si se desea un tamaño menor. Para montar el circuito, debe tener en cuenta que lo primero que debe hacer es soldar los puentes.

La manera más fácil de instalarlos es tomar un solo cable largo, y trenzarlo o "tejerlo" a lo largo de cada una de las líneas de las columnas, después soldar en los puntos indicados y luego recortar el cable sobrante. Después de instalados los puentes, debe soldar los demás componentes; sin embargo, es buena idea que empiece con los de bajo perfil, como las resistencias y luego, con las bases de los integrados y los demás componentes.

En nuestro proyecto, la matriz inicia mostrando la palabra CEKIT, la cual rota hacia arriba, después aparece una cara feliz y luego una esfera dando botes dentro de la matriz. Estos son sólo ejemplos de la infinidad de patrones que se pueden crear; el límite es suimaginación. Usted mismo puede crear sus propios patrones gráficos para lo cual debe modificar el programa. En los sitios comerciales es común encontrar toda clase de motivos, algunos con diseños muy especiales.

Al trabajar con este circuito usted aprende muchas cosas útiles. Un buen programa debe ahorrarle la utilización de uno o varios circuitos externos. El *hardware* y el *software* finales deben comunicarse con tal sincronía y exactitud, que prácticamente se vuelvan una sola unidad.

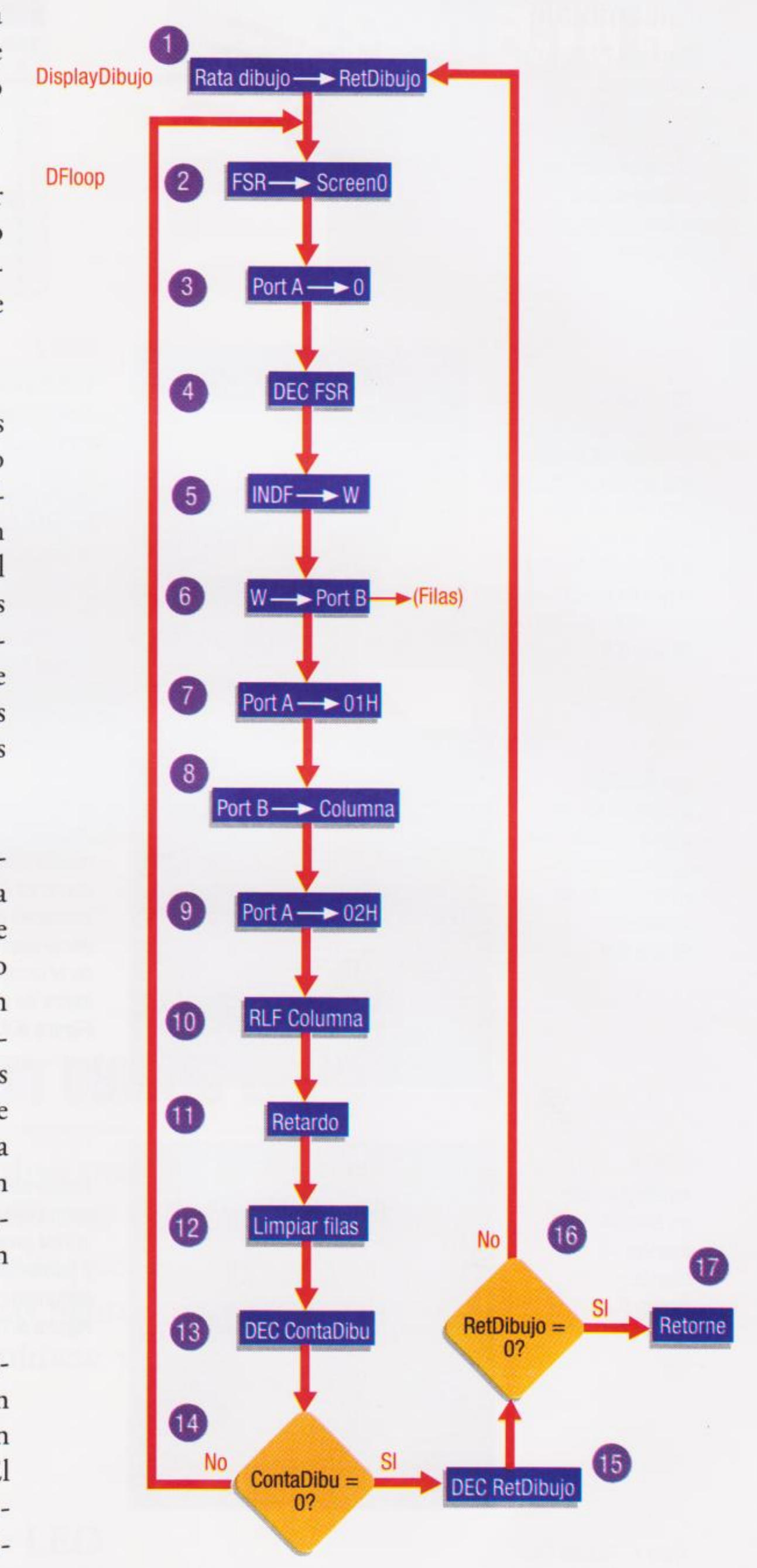


Figura 5.6 *Diagrama de flujo de la rutina DisplayDibujo*

Ensamblaje

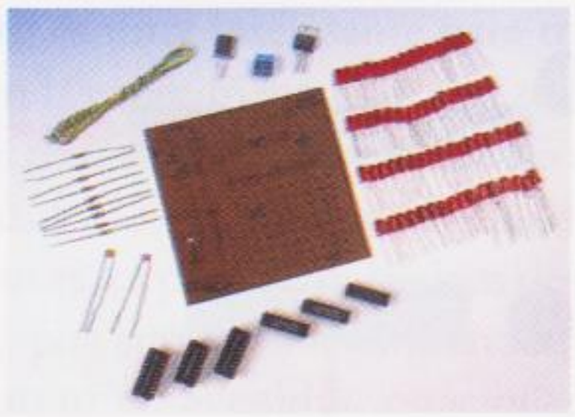
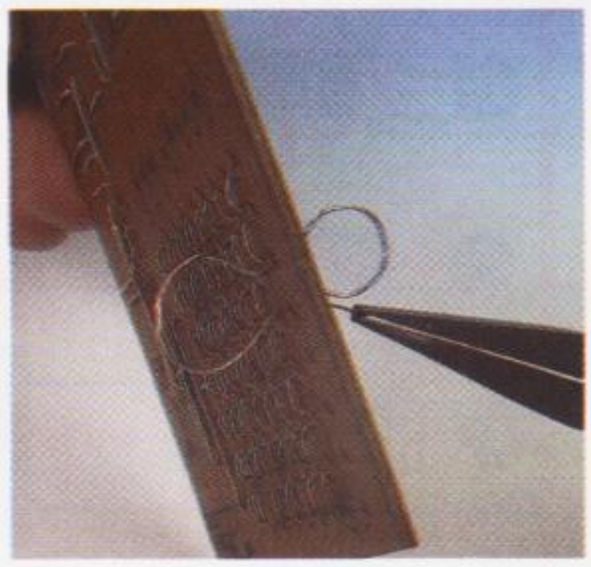


Figura 5.7 *Este es el conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el proyecto.*

LISTA DE MATERIALES		
Cant.	**Ref.**	**Descripción**
8	R1 a R8	Resistencias de 39Ω 1/4W
1	IC1	PIC16F84 programado
2	IC2 - IC3	74HC374
1	IC4	Regulador fijo +5V, 7805
2	C1 y C2	Condensadores cerámicos de 22 pF - 50V
1	Xtal	Cristal de cuarzo de 4MHz
64	LED 1 a LED 64	Diodos LED de 5 mm, rojos
1	K - 208	Circuito impreso CEKIT
2		Bases para circuito integrado de 20 pines
1		Base para circuito integrado de 18 pines
1		Conector de tornillo de 2 pines

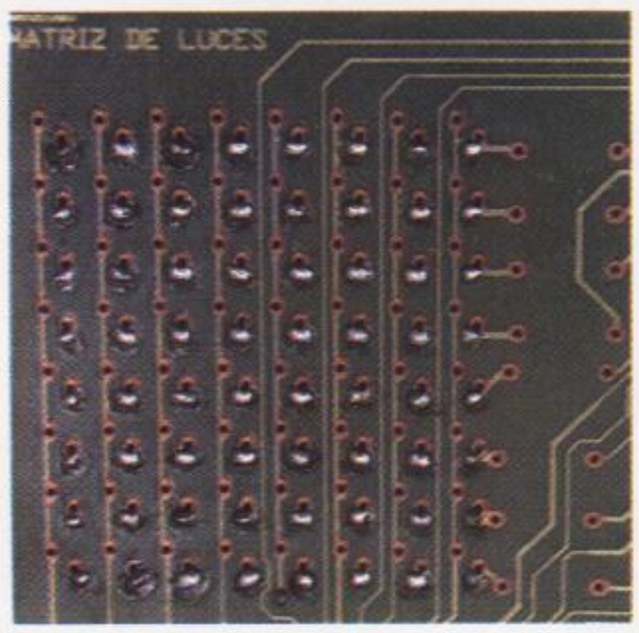
Paso 1

Con un alambre delgado, haga un tejido para conectar los puentes que van debajo de los diodos LED. Esta es la parte más delicada en el ensamblaje de este proyecto y debe hacerse con mucho cuidado.
Figura 5.8



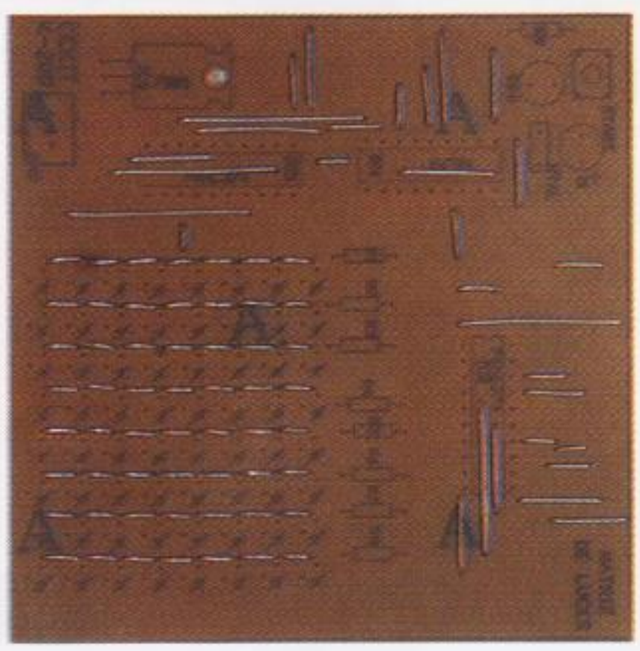
Paso 2

Los puentes anteriores se deben soldar en cada una de sus conexiones, en la cara de las soldaduras.
Figura 5.9



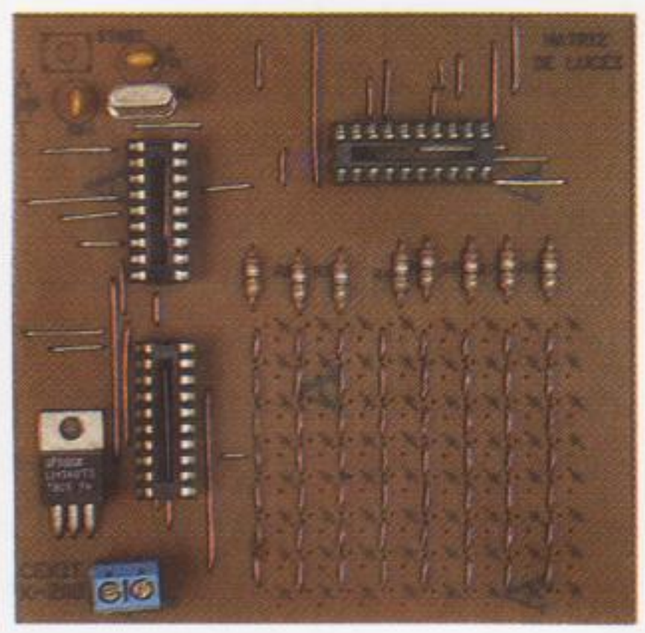
Paso 3

Instale y suelde los demás puentes de alambre.
Figura 5.10



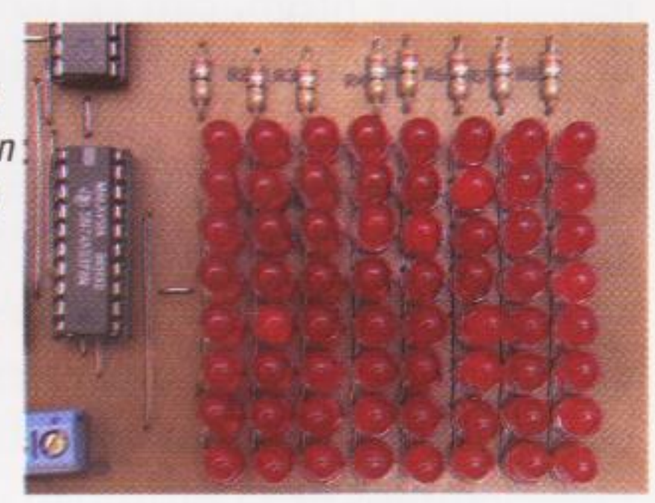
Paso 4

Instale y suelde los demás componentes como resistencias, condensadores, las bases para los circuitos integrados, el cristal, el regulador de voltaje y el terminal para la alimentación.
Figura 5.11



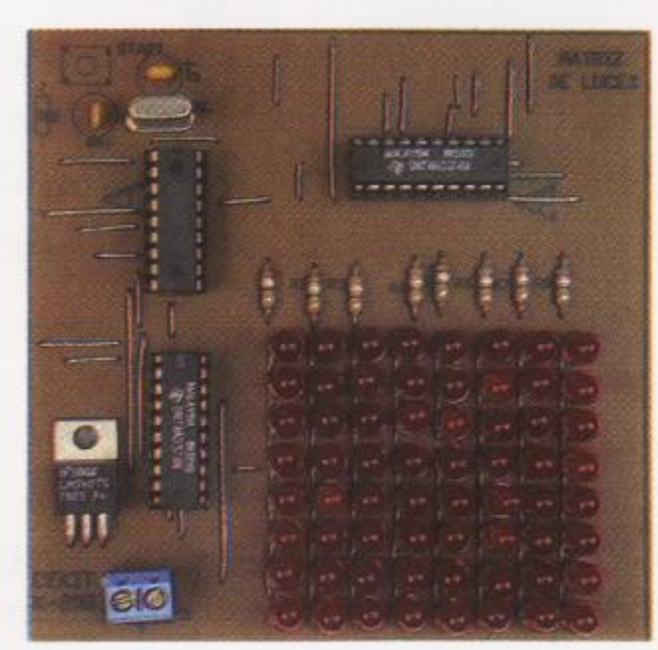
Paso 5

Instale y suelde los 64 diodos LED teniendo en cuenta su posición, ya que si se instalan al contrario, no se encienden. El símbolo en el circuito impreso indica su polaridad.
Figura 5.12



Paso 6

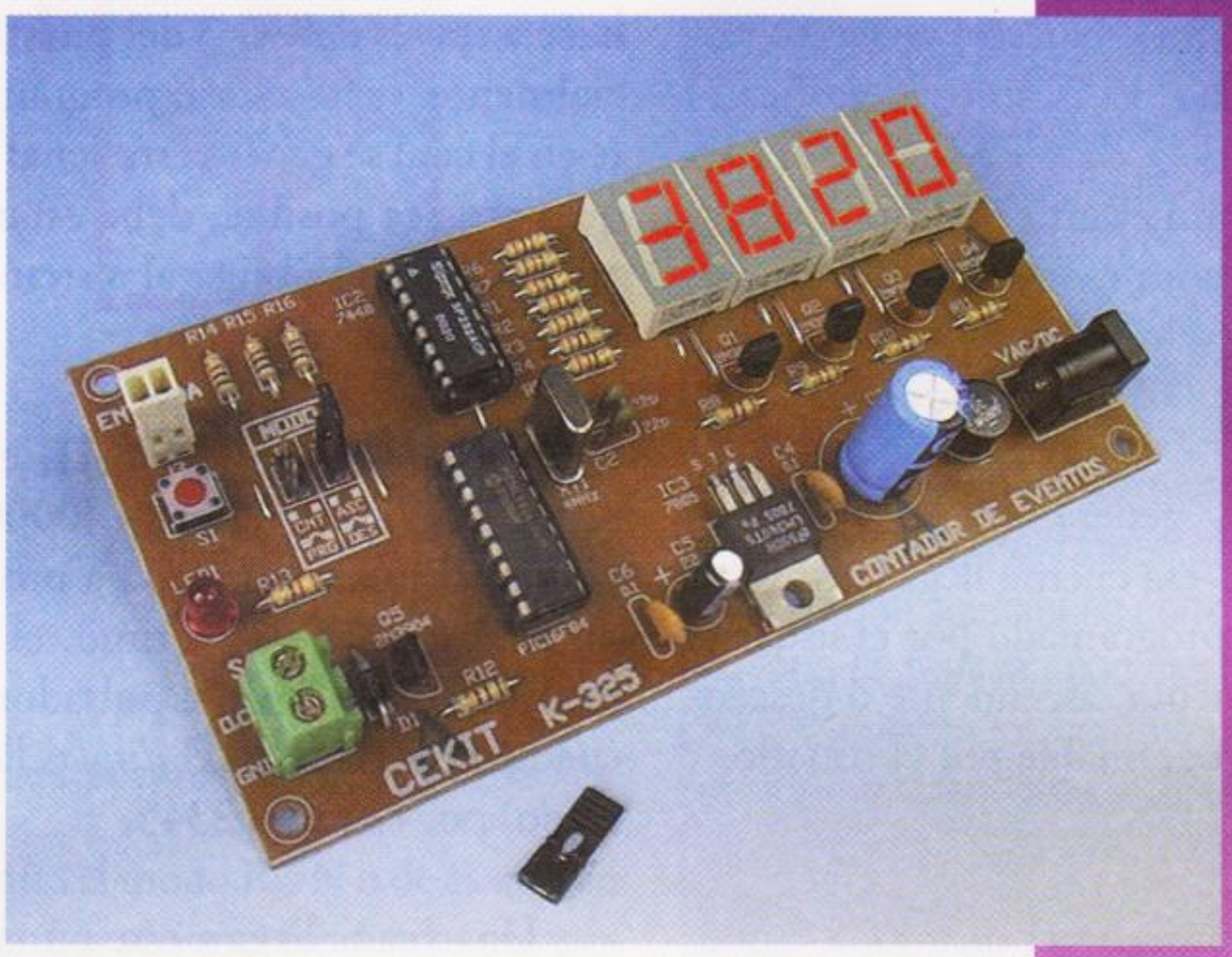
Instale en su base el microcontrolador PIC 16F84 programado y los demás circuitos integrados
Figura 5.13



Paso 7. Prueba final

Para probar el circuito, aliméntelo con una fuente regulada o no, hasta de 12 voltios. En ese momento deben aparecer los efectos mencionados en el texto anterior. Si no es así, revise muy bien las conexiones y verifique que el programa esté grabado en el microcontrolador.

PROYECTO 6



CONTADOR DE EVENTOS

Este circuito cuenta desde 0000 hasta 9999 en forma ascendente o descendente todo tipo de eventos. Aprenderemos a manejar técnicas de **multiplexado** para la presentación de datos numéricos en visualizadores de siete segmentos y a utilizar **macros** en nuestros programas.

Características

- Cuenta en un rango entre 0 y 9999.
- Límite de cuenta programable, tanto en modo ascendente como descendente.
- Generación automática de una señal de alarma cuando la cuenta llega a su límite prefijado.
- Conexión externa para un dispositivo de alarma.

En muchos procesos industriales resulta muy útil un dispositivo que lleve la cuenta de algún evento físico que ocurra con cierta frecuencia, por ejemplo, la cantidad de objetos que pasan por una banda transportadora; y genere al mismo tiempo una señal de control cuando esa cuenta llegue a un valor límite. Nuestro proyecto responde a esta necesidad. El mismo, identificado con la referencia **K-325** de CEKIT, cuenta las veces que se produce un evento externo, detectado mediante un pulsador u otro tipo de dispositivo de entrada, y presenta el resultado en cuatro visualizadores *(displays)* de siete segmentos.

Adicionalmente, genera una señal de control o alarma cuando esa cuenta llega a un valor límite, previamente programado por el usuario. La dirección de la cuenta, ascendente o descendente, es también programable por el usuario.

Funcionamiento

En la **figura 6.1** se muestra el diagrama esquemático del contador de eventos CEKIT K-325. El corazón del sistema es un microcontrolador PIC16F84 (IC1), el cual controla cuatro *displays* de siete segmentos de cátodo común (DISP1-DISP4) a través de un decodificador BCD (IC2). El primer *display* (DISP1) representa el dígito más significativo de la cuenta y el último (DISP4) el menos significativo. Los cátodos de los *displays* se conectan y desconectan de tierra mediante cuatro transistores NPN (Q1-Q4), controlados a su vez por las líneas 4, 5, 6 y 7 del puerto B. Las líneas 0, 1, 2 y 3 del puerto A, por su parte, proporcionan secuencialmente el dato que será visualizado en los *displays*.

Para cada dígito de la cuenta se utiliza un código BCD de cuatro *bits*, donde el MSB (*bit* de mayor peso) es el RA3 (D) y el LSB (*bit* de menos peso) es el RA0 (A). Las cuatro líneas anteriores alimentan un circuito integrado 7448 (IC2), el cual hace la conversión de BCD a siete segmentos. Las resistencias R1 a R7 tienen por objeto limitar la corriente que fluye a través de los segmentos de cada *display* a un valor seguro.

El microcontrolador recibe tres señales de entrada: una por el *bit* 2 del puerto B (pin 8), proveniente de un pulsador (S1), y dos por los *bits* 0 y 1 del mismo puerto (pines 6 y 7), provenientes de dos puentes o *jumpers* (J1 y J2). La salida de alarma o fin de cuenta del sistema se hace a través del *bit* 3 del puerto B (pin 9). Inicialmente, es necesario programar el valor límite en el cual se genera esta señal de control. Para que esto sea posible, debe estar puesto el *jumper* J1, lo cual obliga al sistema a entrar en el modo de programación.

Después de instalado J1, el contador sitúa todos los *displays* en cero (0000) y queda listo para leer el dato que se va a programar como límite. Para fijar este último, el usuario simplemente debe oprimir el pulsador S1 hasta que la cuenta observada en el *display* llegue al valor deseado, por ejemplo 2345.

Una vez se llega a este punto, debe especificarse la dirección de la cuenta. Para ello se utiliza el *jumper* J2. Si este último está conectado, la cuenta se efectúa en sentido descendente a partir del valor fijado, mientras que si está desconectado, se hace en sentido ascendente a partir del mismo valor. Las funciones de los *jumpers* J1 y J2 se resumen en la tabla anexa a la figura 6.1.

En el modo ascendente, la cuenta en los *displays* se inicia con el valor 0000 y se incrementa en una unidad con cada pulso de entrada externo, hasta llegar al valor programado. Cuando esto sucede, la señal de control disponible en la línea RB3 (pin 9) se activa en alto y permanece así por algunos segundos. Hecho esto, la cuenta retorna a 0000 para iniciar una nueva cuenta ascendente.

En el modo descendente, por su parte, la cuenta en los *displays* se inicia con el valor programado previamente y se decrementa en una unidad con cada pulso de entrada externo hasta llegar a cero. Cuando esto sucede, nuevamente se dispara, durante algunos segundos, la salida de control de alarma. Hecho esto, la cuenta re-

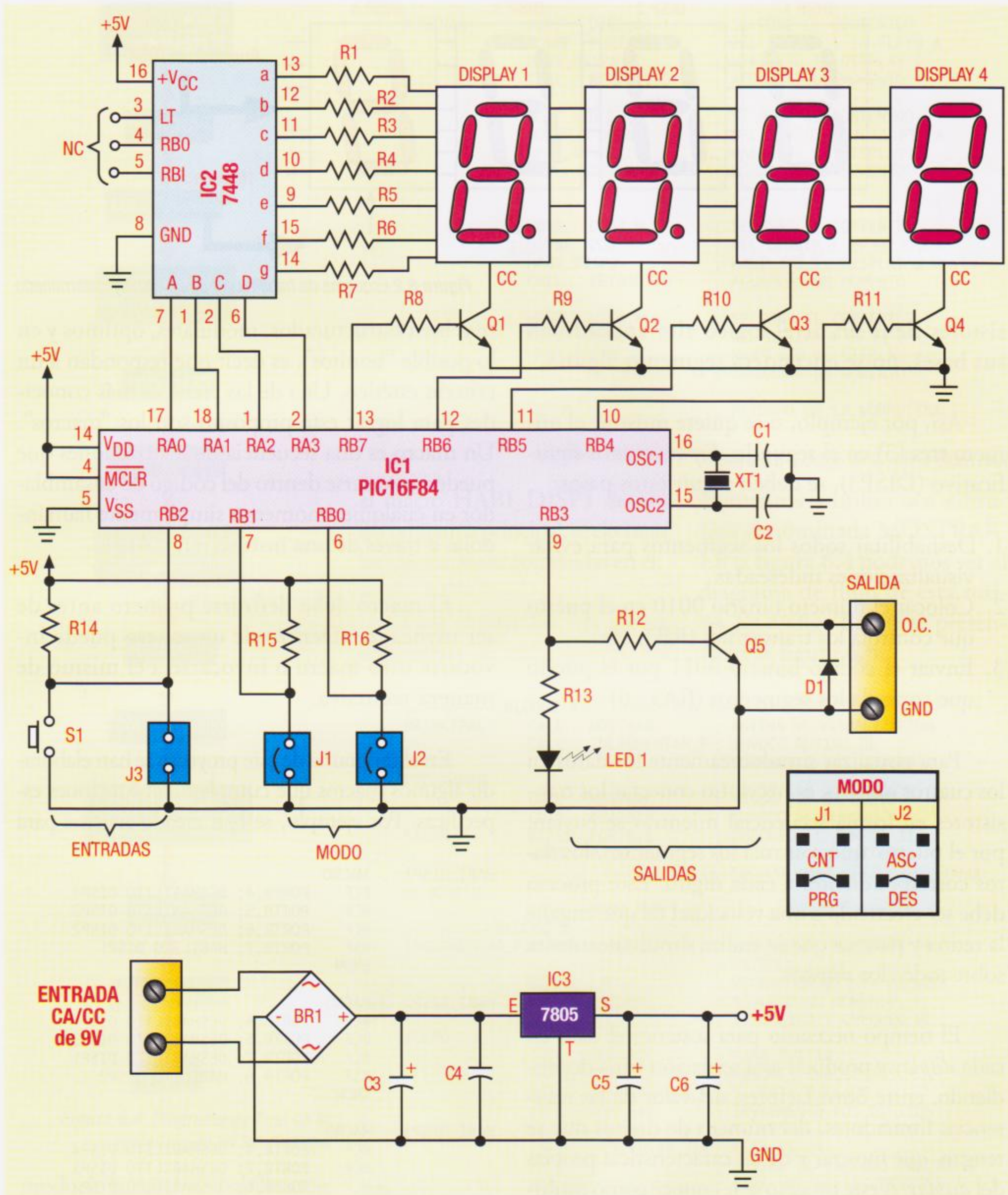


Figura 6.1 Diagrama esquemático del proyecto

torna al valor programado para iniciar una nueva cuenta descendente.

Manejo de los *displays* de siete segmentos

Por la configuración del circuito, es muy fácil deducir que cuando se tiene un nivel lógico alto en la base de cualquier transistor, éste entra en estado de conducción y conecta dinámicamente a tierra la línea de habilitación del *display* correspondiente. Si ninguno de los tran-

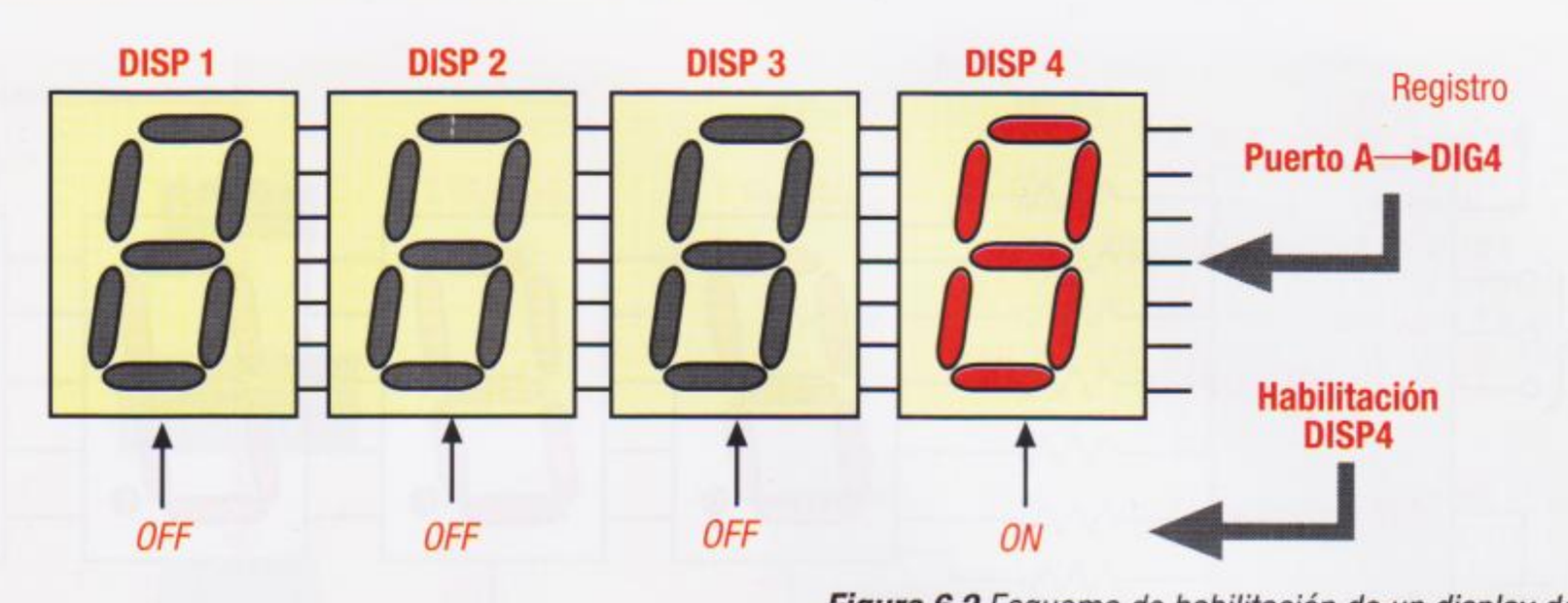


Figura 6.2 *Esquema de habilitación de un display determinado*

sistores tiene un nivel lógico alto aplicado en sus bases, no se encenderá segmento alguno.

Así, por ejemplo, si se quiere mostrar el número tres (3) en el segundo *display* menos significativo (DISP3), se deben seguir estos pasos:

1. Deshabilitar todos los segmentos para evitar visualizaciones indeseadas
2. Colocar el número binario 0010 en el puerto que controla los transistores (RB7...4)
3. Enviar el código binario 0011 por el puerto que controla los segmentos (RA3...0)

Para visualizar simultáneamente los datos en los cuatros *displays*, es necesario conectar los transistores en forma secuencial mientras se envían, por el puerto que controla los segmentos, los datos correspondientes a cada dígito. Este proceso debe ser efectuado a una velocidad tal que engañe la retina y parezca que se realiza simultáneamente sobre todos los *displays*.

El tiempo necesario para sostener el dato en cada *display* y producir esta sensación varía dependiendo, entre otros factores, del valor de las resistencias limitadoras, del número de dígitos que se tengan que mostrar y de las características propias del *display*. A este proceso se le conoce como **multiplexaje** o **multiplexación** y es muy utilizado en los circuitos con microcontroladores.

Descripción del programa
Concepto de macro

Una de las principales metas de todo programador en lenguaje ensamblador es elaborar programas bien estructurados, modulares, óptimos y en lo posible "bonitos", es decir que respondan a un criterio estético. Una de las técnicas más conocidas para lograr este propósito son los "macros". Un **macro** es una secuencia de instrucciones que pueden insertarse dentro del código de ensamblador en cualquier momento simplemente llamándolas a través de una instrucción CALL.

El macro debe definirse primero antes de ser invocado. Dentro de un macro puede invocarse otro macro o invocarse a él mismo de manera recursiva.

En el programa de este proyecto se han elaborado algunos macros que cumplen con funciones específicas. Por ejemplo, se han creado macros para

```
HABI_DISP1    MACRO
              BCF    PORTB,4; DESHABILITO DISP4
              BCF    PORTB,5; DESHABILITO DISP3
              BCF    PORTB,6; DESHABILITO DISP2
              BSF    PORTB,7; HABILITO DISP1
              ENDM

HABI_DISP2    MACRO
              BCF    PORTB,4; DESHABILITO DISP4
              BCF    PORTB,5; DESHABILITO DISP3
              BCF    PORTB,7; DESHABILITO DISP1
              BSF    PORTB,6; HABILITO DISP2
              ENDM

HABI_DISP3    MACRO
              BCF    PORTB,4; DESHABILITO DISP4
              BCF    PORTB,7; DESHABILITO DISP1
              BCF    PORTB,6; DESHABILITO DISP2
              BSF    PORTB,5; HABILITO DISP3
              ENDM

HABI_DISP4    MACRO
              BCF    PORTB,7; DESHABILITO DISP1
              BCF    PORTB,5; DESHABILITO DISP3
              BCF    PORTB,6; DESHABILITO DISP2
              BSF    PORTB,4; HABILITO DISP4
              ENDM
```

Figura 6.3 *Macros de habilitación de cada uno de los displays*

Figura 6.4 *Diagrama de flujo de la visualización en los displays*

```
MOSTRAR     MOVF    DIG1,W        ; SE PONE EL CONTENIDO
            MOVWF   PORTA         ; DEL DIGITO 1 EN ELPTO A
            HABI_DISP1            ; HABILITO EL DISPLAY 1
            CALL    RETARDITO     ; AGUARDAR UN MOMENTO

            MOVF    DIG2,W        ; SE PONE EL CONTENIDO
            MOVWF   PORTA         ; DEL DIGITO 2 EN EL PTO A
            HABI_DISP2            ; HABILITO EL DISPLAY 2
            CALL    RETARDITO     ; AGUARDAR UN MOMENTO

            MOVF    DIG3,W        ; SE PONE EL CONTENIDO
            MOVWF   PORTA         ; DEL DIGITO 3 EN EL PTO A
            HABI_DISP3            ; HABILITO EL DISPLAY 3
            CALL    RETARDITO     ; AGUARDAR UN MOMENTO

            MOVF    DIG4,W        ; SE PONE EL CONTENIDO
            MOVWF   PORTA         ; DEL DIGITO 4 EN EL PTO A
            HABI_DISP4            ; HABILITO EL DISPLAY 4
            CALL    RETARDITO     ; AGUARDAR UN MOMENTO

            RETLW   0             ; RETORNO DE LA SUBRUTINA
```

Figura 6.5 *Código de la rutina MOSTRAR*

el macro HABI_DISP1 dentro del programa principal, se ejecutan las instrucciones contenidas en él.

Para visualizar los datos adecuados en cada uno de los segmentos se utiliza una subrutina denominada MOSTRAR. En la **figura 6.4** podemos ver el diagrama de flujo de esta rutina. Su código fuente se presenta en la **figura 6.5**.

```
;--------- BLOQUE A ------------------
PRINCIPAL   CALL    MOSTRAR       ; RUTINA DE VISUALIZACION
            CALL    INCREMENTAR_P ; INVOCO RUTINA DE
                                  ;INCREMENTO DEL NUMERO A
PROGRAMAR
            CALL    MOSTRAR       ; VISUALIZO DE NUEVO
            GOTO    GUARDAR       ; GUARDO EL NUMERO
SALIR       BTFSS   PORTB,1       ; CHEQUEO EL JUMPER J1
            GOTO    PRINCIPAL     ; SI NO SE HA QUITADO
                                  ; REGRESO PARA SEGUIR AUMENTANDO
EL NUMERO
            GOTO    MODO_RUN
;---------BLOQUE B-----------------
INCREMENTAR_P BTFSC PORTB,2       ; CHEQUEO EL PULSADOR DE
                                  ; ENTRADA
            RETLW   0             ; RETORNO SI NO ESTA PULSADO
            CALL    RETARDO       ; AGUARDO UN MOMENTO,
                                  ; VISUALIZO Y CHEQUEO DE
NUEVO
            CALL    MOSTRAR       ; PARA EVITAR
            BTFSC   PORTB,2       ; REBOTES INDESEADOS.
            GOTO    INCREMENTAR_P ; SI ESTA PULSADO
            CALL    MOSTRAR       ; INVREMENTO LOS DIGITOS
                                  ; Y  VISUALIZO DE NUEVO.
            CALL    CALCULAR_INC  ; ESTA ES LA
                                  ; RUTINA DE CALCULO EXACTO DE
        CALL MOSTRAR              ; CADA UNO DE LOS DIGITOS
            RETLW   0

GUARDAR     MOVF    DIG4,W        ; AQUI ALMACENO CADA UNO
            MOVWF   DIG4_P        ; DE LOS DIGITOS EN
            MOVF    DIG3,W        ; REGISTROS AUXILIARES
            MOVWF   DIG3_P        ; UBICADOS EN OTRAS
            MOVF    DIG2,W        ; POSICIONES EN RAM.
            MOVWF   DIG2_P
            MOVF    DIG1,W
            MOVWF   DIG1_P
            GOTO    SALIR
```

A

B

Figura 6.6 *Subrutina de programación del valor límite*

producir la señal de habilitación de cada uno de los *displays*. Cuando se pretende visualizar un dato en algún *display* determinado, se debe habilitar ese *display* y deshabilitar los demás, **figura 6.2**. Los macros de la **figura 6.3** hacen esa tarea, de manera, que invocando

Figura 6.7 *Diagrama de flujo de la subrutina de programación*

La rutina anterior es en realidad muy sencilla. Su función consiste en tomar el contenido de los registros DIG1, DIG2, DIG3 y DIG4, los cuales contienen el dato completo que se debe entregar a cada uno de los *displays*, y enviar el código de cada dígito, uno a la vez, al puerto A mientras habilita el *display* correspondiente. Para ello recurre al macro asociado a ese *display*. En la **figura 6.6** observamos la subrutina con la cual se programa el valor límite del contador y en la **figura 6.7** se muestra su respectivo diagrama de flujo.

En esta subrutina se lee el pulsador de entrada conectado al *bit* 2 del puerto B (bloque B del código). En caso de confirmarse una entrada externa, se invoca la subrutina **CALCULAR_INC**, la cual se encarga de asignar a cada uno de los registros DIG1 hasta DIG4, su valor exacto.

Hecho esto, mediante la subrutina **GUARDAR**, los valores de los dígitos se almacenan en

```
;--------BLOQUE C ----------------
ASC_DES       MACRO                ; MACRO QUE DETECTA
              BTFSC  PORTB,0       ; EL SENTIDO DEL
C             GOTO   ASCEND        ; CONTEO MEDIANTE EL
DESCEN        BSF    BANDERA,0     ; CHEQUEO DE EL
              REST_DIGITOS         ; BIT 0 DEL PUERTO B
              GOTO   OUT           ; SEGUN ESO LOS DIGITOS
ASCEND        BCF    BANDERA,0     ;INICIAN EN 0 O CON
              INICIAR_DIGITOS      ; EL VALOR PROGRAMADO
OUT           NOP
              ENDM
;---------BLOQUE D-----------------
MODO_RUN      ASC_DES              ;SE DETERMINA SI EL CONTEO ES
              BTFSS  BANDERA,0     ; ASCENDENTE O DESCEN.
D             GOTO   INCREMENTAR   ; SEGUN EL BIT 0 DE
                                   ; BANDERA, SALTO A
              GOTO   DECREMENTAR   ; INCREM. O DECREM.
;--------BLOQUE E -------------------
INCREMENTAR   BTFSC  PORTB,2       ; CHEQUEO DE EL
                                   ; PULSADOR DE ENTRADA
              GOTO   NORMAL        ; SE AGUARDA UN
RETARDO                            ; PARA
              CALL   RETARDO       ; EVITAR LECTURA DE
              CALL   MOSTRAR       ; REBOTES DEL PULSADOR.
E             BTFSC  PORTB,2       ; SI SE CORROBORA LA
                                   ; LECTURA, SE INVOCA LA
              GOTO   INCREMENTAR   ; SUBRUTINA QUE HACE
              CALL   MOSTRAR       ; UN INCREMENTO DE UNA
              CALL   CALCULAR_INC  ; UNIDAD EN LOS
                                   ; VALORES DE LOS DISPLAYS
              CALL   MOSTRAR       ; SE FINALIZA MOSTRANDO
              GOTO   NORMAL        ; LOS DATOS.
DECREMENTAR   BTFSC  PORTB,2       ; SE LEE EL PULSADOR DE
                                   ; ENTRADA.
              GOTO   NORMAL        ; SE AGUARDA UN RETARDO
              CALL   RETARDO       ; PARA EVITAR LECTURA DE
              CALL   MOSTRAR       ;REBOTES DEL PULSADOR.
              BTFSC  PORTB,2       ;SI SE CORROBORA LA LEC-
              GOTO   DECREMENTAR   ;TURA, SE INVOCA LA
                                   ;SUBRUTINA QUE
              CALL   MOSTRAR       ; HACE UN DECREMENTO DE
                                   ; UNA UNIDAD EN LOS VALO-
              CALL   CALCULAR_DEC  ;RES DE LOS DISPLAYS
              CALL   MOSTRAR       ; SE FINALIZA MOSTRANDO
                                   ; LOS DATOS
              GOTO   NORMAL
;---------BLOQUE F ---------------------
NORMAL        CALL   MOSTRAR       ; EN ESTA SUBRUTINA SE
                                   ; HACE UNA COMPARACION
ALARM         BTFSS  BANDERA,0     ; ENTRE EL NUMERO
              GOTO   WE            ; ACTUAL QUE SE ESTA
F                          ; ORIGINANDO CON EL PULSADOR
              GOTO   WE1           ; CON EL VALOR PROGRAMADO.
WE            CALL   ALARMA        ;SI SON IGUALES SE
                                   ; ACTIVAN LAS RUTINAS DE
              GOTO   INCREMENTAR   ; ALARMA. HAY UNA
WE1           CALL   ALARMA2       ; RUTINA PARA EL
                                   ; INCREMENTO Y OTRA
              GOTO   DECREMENTAR   ;PARA EL DECREMENTO
```

Figura 6.8 *Código de la subrutina que ejecuta el funcionamiento normal del sistema*

registros auxiliares para ser utilizados más adelante con el fin de verificar cuando debe generarse la señal de control. En el bloque A del código se efectúa un ciclo continuo. Mediante este ciclo se muestran los datos programados actualmente, se determina si hay entrada externa y se guardan los datos nuevos. Dicho ciclo se repite indefinidamente mientras el *jumper* J1 no sea removido. Cuando se retira

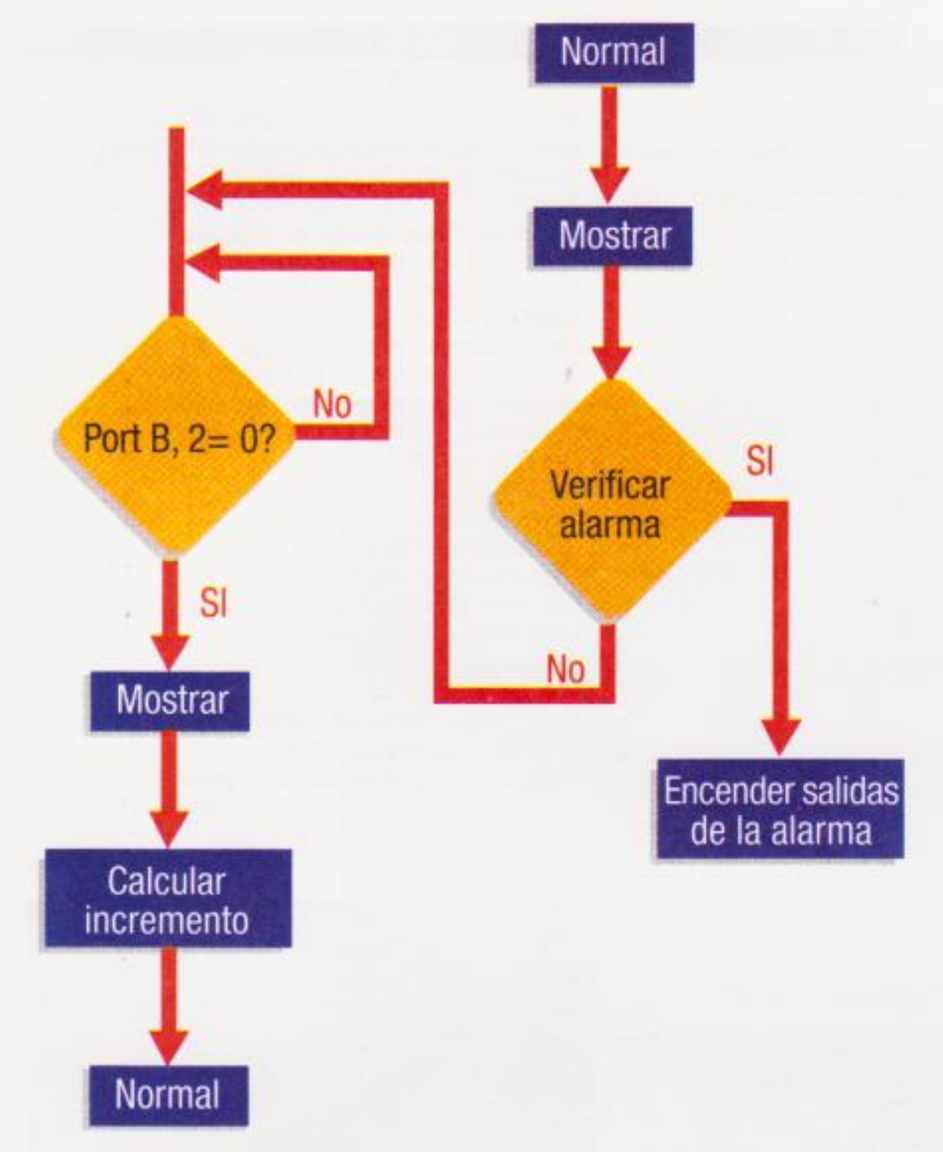


Figura 6.9 *Diagrama de flujo de la operación en modo RUN*

J1, el programa entra al modo RUN o de ejecución normal del sistema. Las acciones efectuadas por esta parte del programa podemos apreciarlas en el código de la **figura 6.8** y en el diagrama de la **figura 6.9**.

En el **bloque D** se lee el estado de J2 para determinar si la cuenta va a ser efectuada en forma ascendente o descendente. En el **bloque C** se hace un macro el cual, según el estado de J2, hace que los valores de los registros DIG1 a DIG4 adopten el valor 0000 en caso de cuenta en modo ascendente, o el valor programado previamente si se ha seleccionado el modo descendente.

En el **bloque E** del programa se monitorea permanentemente el estado de la entrada externa para hacer la cuenta. Si, en un momento dado, se lee una entrada válida por este pin, se invoca la subrutina CALCULAR_INC o CALCULAR_DEC para calcular el nuevo valor de los registros que guardan el valor de cada dígito. A partir de este punto se entra en un nuevo ciclo, en el cual se incrementan o decrementan los valores de los dígitos, se muestran en los *displays* y se verifica si han llegado al valor límite para disparar la alarma o señal de control, **bloque F** del programa.

PARA SABER MÁS

¿Qué es un macro?

Por definición, un "macro" se refiere a un elemento que se compone de varias partes más pequeñas. Esta misma definición se aplica a los macros del ensamblador, los cuales son conjuntos de directivas e instrucciones definidas por el usuario y que pueden ser incluidos dentro del código al invocar el macro.

Los macros pueden escribirse de tal forma que acepten argumentos, haciéndolos muy flexibles. Entre otras, el uso de macros tiene las siguientes ventajas:

- Proporcionan un alto nivel de abstracción, haciendo los programas más comprensibles
- Ofrecen soluciones consistentes a funciones que se realizan con cierta frecuencia
- Simplifican los cambios
- Facilitan la prueba y depuración del programa

Las aplicaciones más frecuentes de los macros son:

- Construcción de tablas complejas
- Creacción de códigos usados con frecuencia
- Realización de operaciones complejas

Definición de un macro

La definición de un macro se divide en tres partes: la cabecera, el cuerpo y la terminación. La cabecera de un macro posee la siguiente sintaxis:

<nombre> MACRO [<arg> [, <arg>]]

donde **<nombre>** es el nombre del macro y **<arg>** son los parámetros que se le entregan al macro desde el punto donde se invoca. El cuerpo del macro empieza inmediatamente después de la cabecera y continúa hasta que se le da la terminación. El cuerpo del macro se compone de una serie de líneas de código que cumplen con una determinada labor.

La directiva **ENDM** termina la definición del macro. Ella debe existir antes de que se encuentre otro nuevo macro.

Ensamblaje

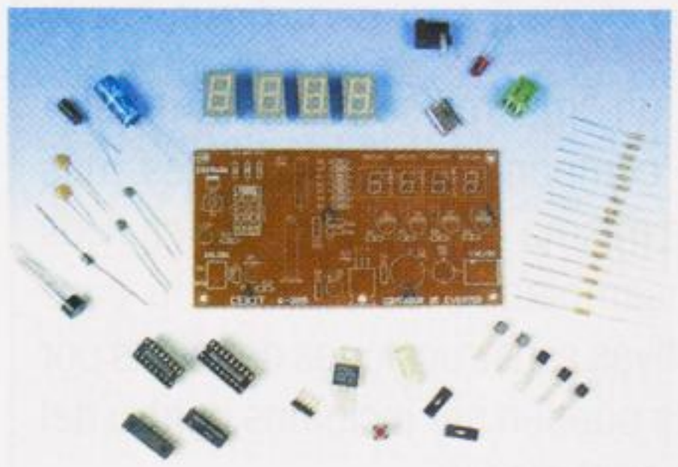
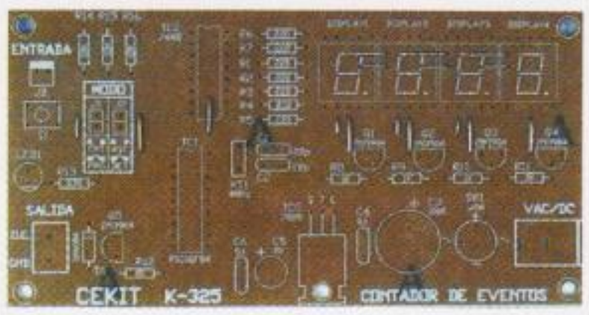


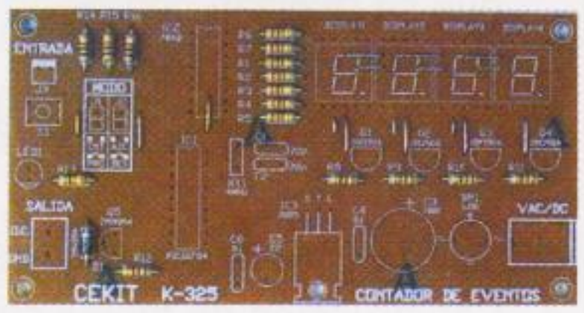
Figura 6.10
Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el proyecto.

LISTA DE MATERIALES		
Cant.	**Ref.**	**Descripción**
1	IC1	Microcontrolador PIC16F84 programado
1	IC2	Decodificador de siete segmentos cátodo común, 7448
1	IC3	Regulador fijo de +5V -1A, 7805
5	Q1 a Q5	Transistores de propósito general NPN, 2N3904
1	BR1	Puente rectificador circular de 1A, W06M
4	DISP1 a DISP4	*Displays* de siete segmentos cátodo común
1	D1	Diodo de propósito general 1N4004
1	LED1	Diodo LED rojo de 5mm
2	C1 y C2	Condensadores cerámicos de 22pF - 50V
1	C3	Condensador electrolítico de 1000uF - 25V
2	C4 y C6	Condensadores cerámicos de 0.1uF - 50V
1	C5	Condensador electrolítico de 22uF - 16V
1	XT1	Cristal de quarzo de 4MHz
7	R1 a R7	Resistencias de 47Ω - 1/4W
5	R8 a R12	Resistencias de 1k - 1/4W
1	R13	Resistencia de 330Ω - 1/4W
3	R14, R15 y R16	Resistencias de 10k - 1/4W
1	S1	Pulsador pequeño de 4 pines, para circuito impreso
2	J1 y J2	*Jumpers* para circuito impreso
1	K-325	Circuito impreso CEKIT
1		Conector CA/CC
1		Conector en línea M/H de 2 pines
1		Conector de tornillo de 2 pines
1		Base para circuito integrado de 18 pines
1		Base para circuito integrado de 16 pines
1		Conector macho tipo cerca de 4 pines

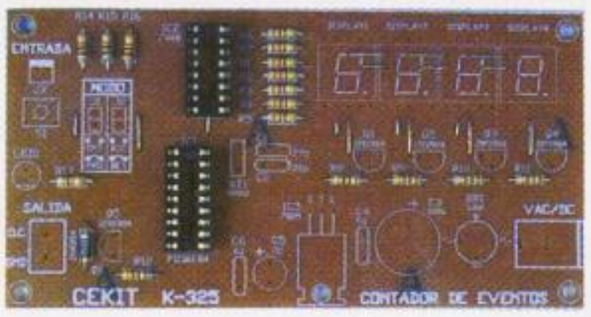
Paso 1
Con un alambre delgado, instale y suelde los puentes marcados en el circuito impreso.
Figura 6.11



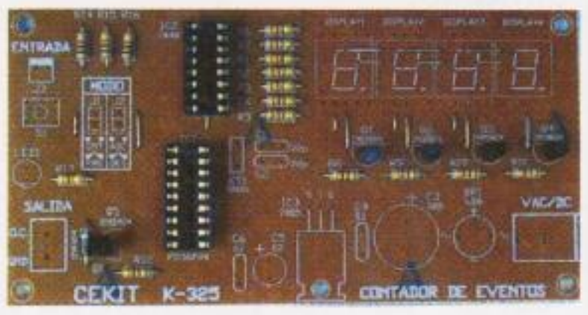
Paso 2
Luego instale y suelde las resistencias (16) y el diodo D1.
Figura 6.12



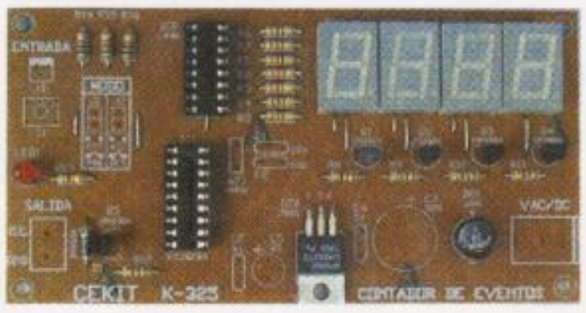
Paso 3
Instale y suelde las bases o sockets para los dos circuitos integrados
Figura 6.13



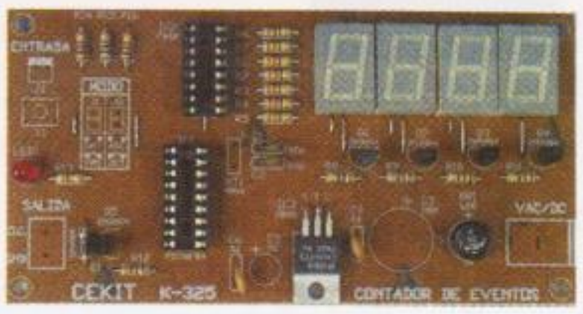
Paso 4
Instale y suelde los cinco transistores NPN observando su posición correcta
Figura 6.14



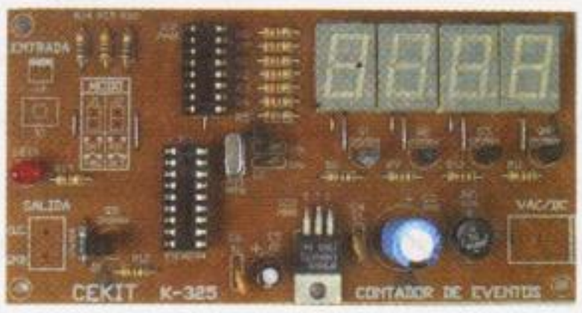
Paso 5
Instale y suelde los cuatro displays, el LED1, el regulador IC3 y el puente rectificador BR1.
Figura 6.15



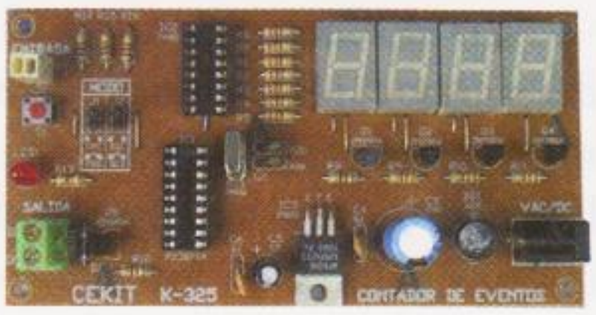
Paso 6
Instale los cuatro condensadores cerámicos C1, C2, C4 y C6
Figura 6.16



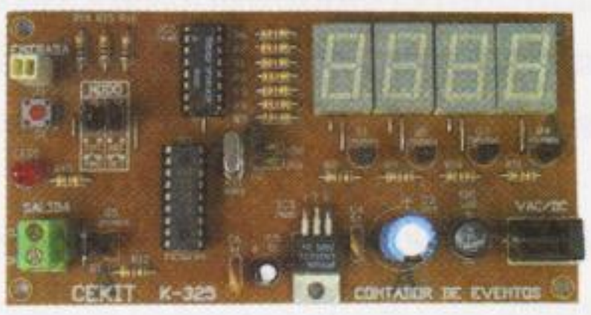
Paso 7
Instale el cristal XT1 y los condensadores electrolíticos C3 y C5 observando su posición
Figura 6.17

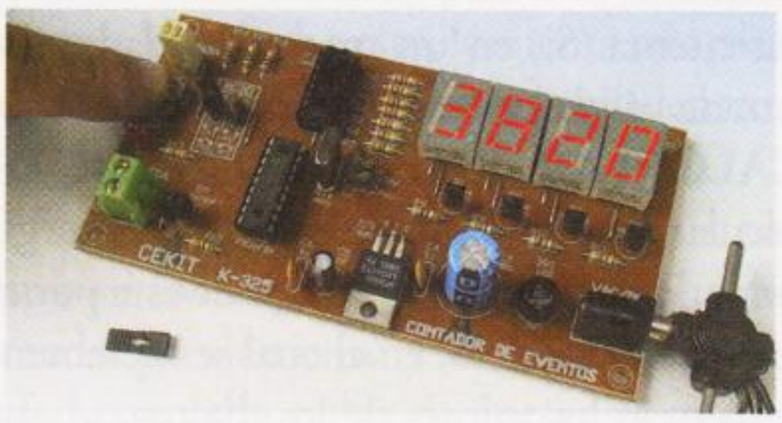


Paso 8
Instale los terminales para los puentes J1 y J2, el conector J3, el pulsador S1 y el conector de alimentación
Figura 6.18



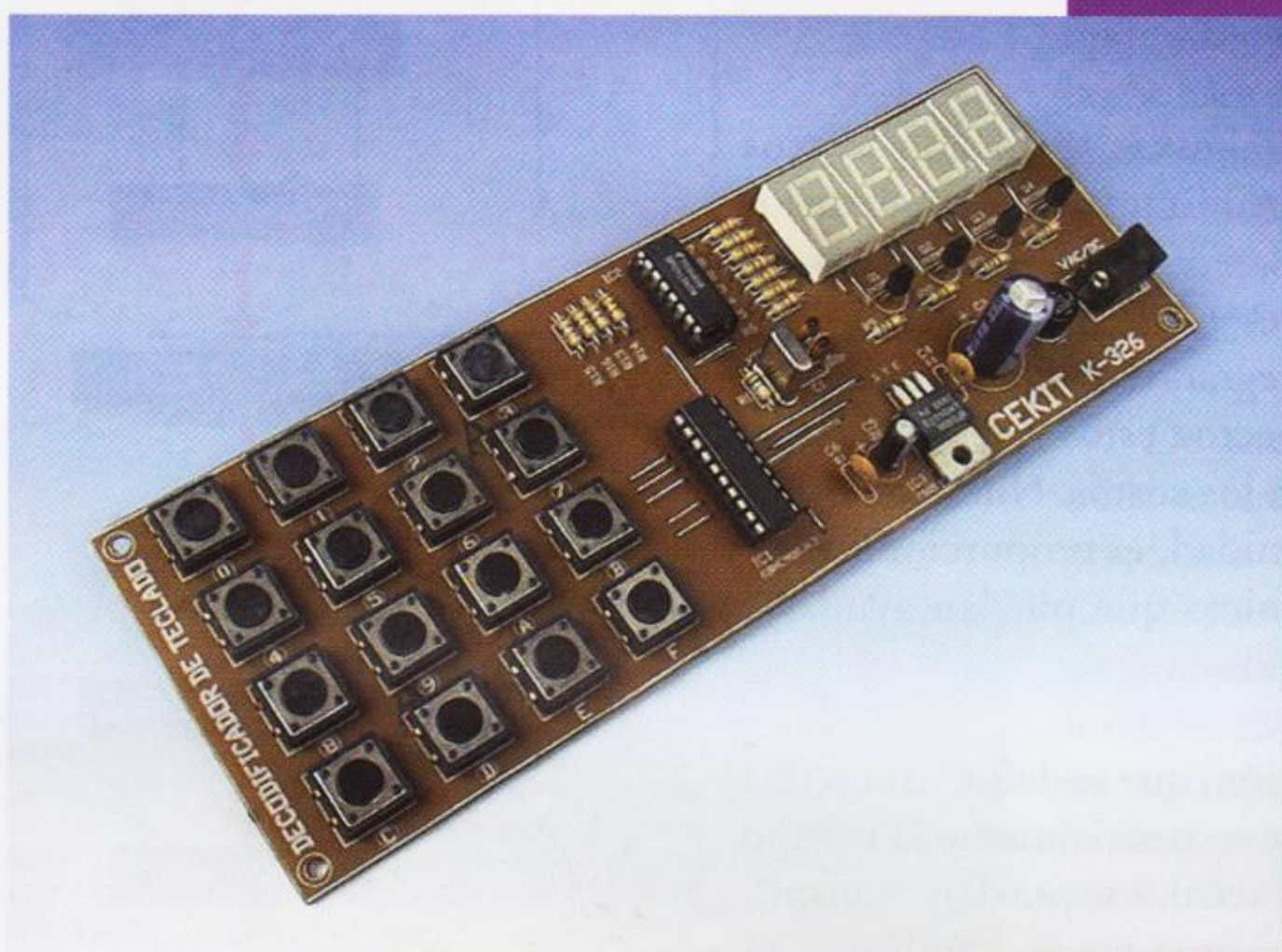
Paso 9
Instale los circuitos integrados en sus bases y los puentes J1 y J2
Figura 6.19





Paso 10. Prueba final
Alimente el circuito con una fuente regulada o no, hasta de 12 voltios. Conecte los puentes J1 para activar el modo de programación y J2 para establecer el modo de la cuenta. Con el pulsador S1 seleccione el número que se va a contar. Retire el puente J1 y presione S1 repetidamente, en ese momento se debe incrementar o decrementar el número según el modo seleccionado. Si no es así, revise muy bien las conexiones y verifique que el programa esté bien grabado en el microcontrolador.

PROYECTO 7



DECODIFICADOR DE TECLADO HEXADECIMAL

En este proyecto se manejan simultáneamente un teclado y cuatro *displays*. Existen muchas aplicaciones del mundo real donde se requiere manejar estos dos elementos, por eso la importancia de entender bien este concepto.

Características

- Lectura de teclado de 4x4
- Visualización de los datos en 4 *displays* de siete segmentos
- Utiliza el microcontrolador Motorola 68HC908JK3

Uno de los problemas que enfrentan con frecuencia los diseñadores y experimentadores de sistemas electrónicos con microcontroladores es que algunas veces las líneas de entrada/salida que tienen disponibles en un dispositivo, parecen no ser suficientes para una aplicación determinada.

En ocasiones, algunas técnicas y trucos pueden ayudar a optimizar las funciones de los microcontroladores, reduciéndose el tamaño de los circuitos impresos y la necesidad de conseguir circuitos integrados con mayor número de líneas I/O; aspectos estos que, necesariamente, disminuyen los costos. Nuestro propósito en esta oportunidad, es proporcionarle algunas de estas técnicas que puedan ayudar a optimizar los diseños.

La multiplexación, que se define como una forma de compartir secuencialmente el tiempo para que dos o más señales se puedan transmitir a la vez por un mismo medio conductor, es sin duda una gran herramienta (y en ocasiones la única) para conseguir un mejor aprovechamiento de un dispositivo. Esta técnica se utiliza ampliamente en comunicaciones (aún más con el auge de la fibra óptica). Usted la utilizará para la lectura de teclados y la visualización a través de *displays* de siete segmentos.

En este proyecto se hará la lectura de un teclado de cuatro filas y cuatro columnas y podrá visualizarse el resultado en cuatro *displays* de siete segmentos. Los *displays* muestran inicialmente el número 0000 y luego muestran los datos que se van introduciendo por el teclado. Como son cuatro *displays*, hay que entrar cuatro datos, cada uno de ellos se ingresa por el teclado y se da un *enter* para indicar que se ha aceptado el número y que se puede pasar al siguiente *display*. En el teclado cualquier letra entre la "A" y la "E" sirve para dar la señal de *enter* y la letra "F" sirve para borrar los *displays*. En otras palabras la rutina es la que se muestra en la **figura** 7.1.

Figura 7.1 *Rutina general para la entrada y visualización de los datos*

Funcionamiento

En la **figura** 7.2 puede apreciar el diagrama esquemático del circuito en el cual vemos que se utiliza un microcontrolador Motorola 68HC908JK3. La salida de los datos hacia los *displays* se hace por los *bits* 4, 5, 6 y 7 del puerto B, correspondientes a los pines 11, 8, 7 y 6, respectivamente. Los *bits* 1, 2 y 3 del puerto B y el *bit* 6 del puerto D se configuran como salidas y van conectados a las filas del teclado.

Manejo de teclados

La clave para manejar este tipo de teclados consiste en enviar por las líneas de salida sólo un cero por vez; por ejemplo, si enviamos un cero por la línea PTD6 (pin 10), cuando oprimimos una tecla en la primera fila (el 0, 1, 2 ó 3), un nivel lógico bajo se reflejará en el pin correspondiente de las líneas de entrada (PTD2, PTD3, PTD4 y PTD5 respectivamente); así, si se encuentra un nivel lógico bajo en la línea PTD5 se puede concluir que la tecla presionada fue el dígito 3.

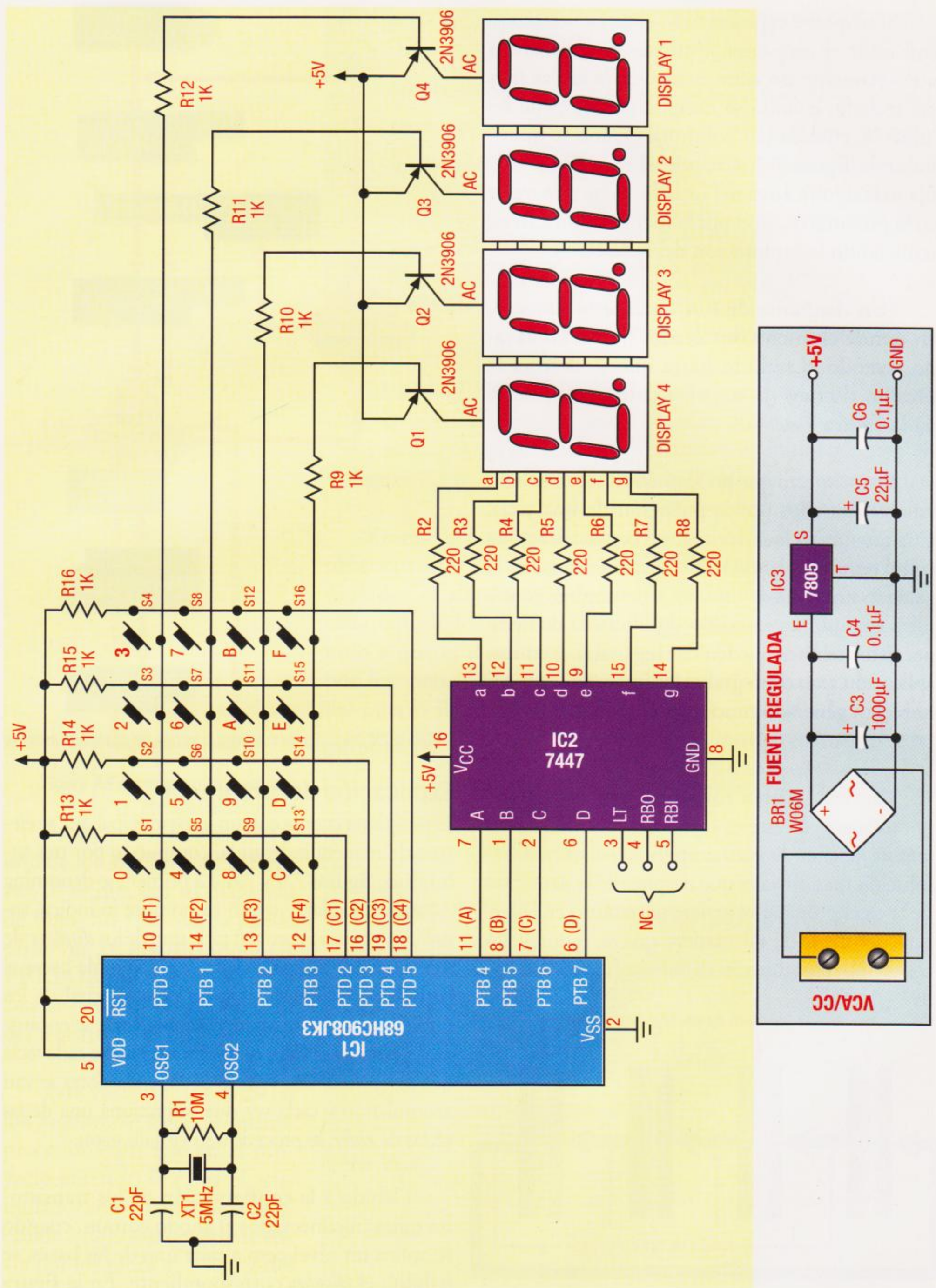


Figura 7.2 *Diagrama esquemático*

Si se quiere explorar todo el teclado, bastará con rotar el cero circularmente, de tal manera que solamente un cero se encuentre en las filas del teclado, cuando se hacen las lecturas de las líneas de entrada (las columnas) como se muestra en la **figura** 7.3. Cuando el cero llegue a la fila más significativa del teclado, debe reingresar en la próxima ocasión por la menos significativa, reiniciando la exploración del teclado.

Un diagrama de flujo para este proceso, en donde el microcontrolador queda enclavado leyendo el teclado hasta que se detecta la presión de uno de sus elementos, se muestra en la **figura** 7.4.

Otro aspecto que no se puede olvidar, son los rebotes causados por la pulsación de una tecla. Cuando una tecla se oprime, sus contactos actúan como resortes, y la unión eléctrica no es estable, se generan una serie de uniones y desuniones mecánicas durante un intervalo significativo de tiempo. Estos rebotes pueden dar lugar a que, en una aplicación real, el programa los interprete como si hubieran generado muchas pulsaciones, si es que no se toman los correctivos del caso.

Para ello existen soluciones por *hardware* y por *software* ; en este caso consideramos más interesante la segunda pues simplifica el diseño. Allí la solución más obvia es que después de la detección de la tecla pulsada, se genere un retardo en la lectura del dato, de tal manera que se ignoren los contactos subsiguientes debidos a los rebotes.
▯

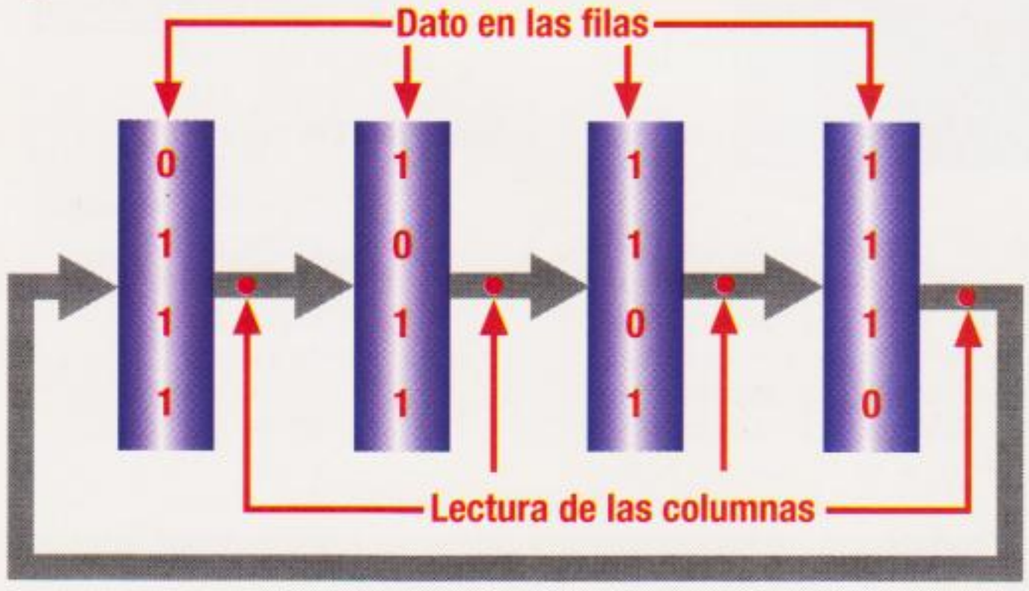


Figura 7.3 *Secuencia para la lectura de un teclado matricial*

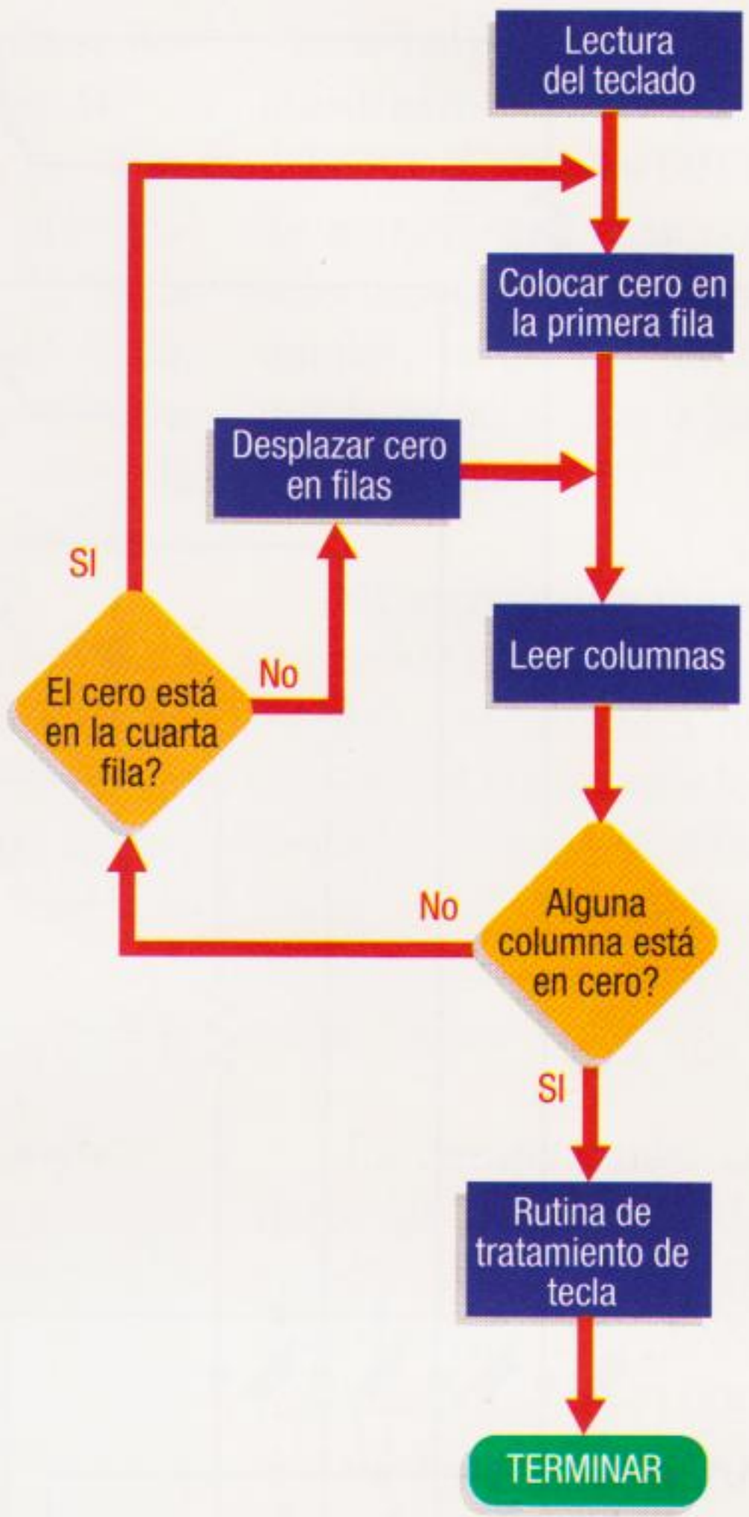


Figura 7.4 *Diagrama de flujo para la lectura de un teclado matricial*

Explicación del programa

El programa cuenta con un lazo principal que se ejecuta de manera indefinida, compuesto por tres subrutinas, **figuras** 7.5 y 7.6. La primera se denomina **ActualiceDisplay** y como su nombre lo indica, actualiza y muestra datos en cada uno de los *displays* de siete segmentos; **LeaTeclado** se encarga de hacer el barrido de las columnas para detectar cual de los pulsadores se ha activado y **AnaliceTecla** se encarga de determinar el valor que se ha asignado a cada tecla que se oprima. Los valores de los pulsadores se van acumulando y cada vez que se presiona una de las teclas de *enter*, se procede a su visualización.

Debido a la configuración de los transistores que van conectados al ánodo común, cuando se aplica un nivel cero a cada una de las bases, se habilita el *display* correspondiente. En la **figura** 7.7 se muestra el código de la habilitación de cada

Figura 7.5 *Diagrama de flujo del lazo principal del programa*

uno de los *displays* y en la **figura 7.8** se muestra su diagrama de flujo. En este código puede observar la instrucción **bclr PB1,PORTB** por medio de la cual se lleva un cero al *bit* PB1 del registro PORTB.

Para visualizar los datos en los *displays*, se sigue el mismo procedimiento que en el proyecto anterior; se carga el acumulador con el dato que corresponde a cada *display*, este se envía por los pines 4, 5, 6 y 7 del puerto B y luego se habilita el *display* correspondiente.

```
**** Inicio del programa al dar RESET *********

Start:

      jsr InitDisplay          ;rutina de inicializacion de la tarea
                               ;de display
      jsr InitKey              ;rutina de inicializacion de la tarea
                               ;de teclado
      jsr InitAnalice          ;rutina de inicializacion de la tarea
                               ;de analisis

loopPpal                       ;loop principal que invica todas las
                               ;tareas del sistema
      jsr ActualiceDisplay     ;Tarea de display
      jsr LeaTeclado           ;Tarea teclado
      jsr AnaliceTecla         ;Tarea de analisis
      bra loopPpal
```

Figura 7.6 *Código fuente del lazo principal del programa*

El código de esta operación se muestra en la **figura 7.9** y el diagrama de flujo en la **figura 7.10**.

En este procedimiento se observa como se lleva al acumulador el dato asignado a cada uno de los *displays* Debido a que el dato debe salir por los cuatro *bits* más altos del puerto B, se debe enmascarar el *nibble* bajo e intercambiar de posición los *nibbles*

Enmascarar: técnica con la cual se utiliza una "máscara" o rutina de inhibición con el fin de alterar o aislar ciertas posiciones de *bit*.

Nibble: grupo de 4 *bits* (medio *byte*) operado como una unidad.

```
EnableDisp1
          bclr PD6,PORTD  ; Configuro el bit 6 del puerto D
          bset DDRD6,DDRD ; (pin 10) como salida y lo pongo en cero.
          RTS
EnableDisp2
          bclr PB1,PORTB  ; Configuro el bit 1 del puerto B
          bset DDRB1,DDRB ; (pin 14) como salida y lo pongo en cero.
          RTS
EnableDisp3
          bclr PB2,PORTB  ; Configuro el bit 2 del puerto B
          bset DDRB2,DDRB ; (pin 13) como salida y lo pongo en cero.
          RTS
EnableDisp4
          bclr PB3,PORTB  ; Configuro el bit 3 del puerto B
          bset DDRB3,DDRB ; (pin 12) como salida y lo pongo en cero.
          RTS
DisabDisp1
          bset PD6,PORTD  ; Configuro el bit 6 del puerto D
          bset DDRD6,DDRD ; (pin 10) como salida y lo pongo en uno.
          RTS
DisabDisp2
          bset PB1,PORTB  ; Configuro el bit 1 del puerto B
          bset DDRB1,DDRB ; (pin 14) como salida y lo pongo en uno.
          RTS
DisabDisp3
          bset PB2,PORTB  ; Configuro el bit 2 del puerto B
          bset DDRB2,DDRB ; (pin 13) como salida y lo pongo en uno.
          RTS
DisabDisp4
          bset PB3,PORTB  ; Configuro el bit 3 del puerto B
          bset DDRB3,DDRB ; (pin 12) como salida y lo pongo en uno.
          RTS
```

Figura 7.7 *Código para la habilitación de los displays*

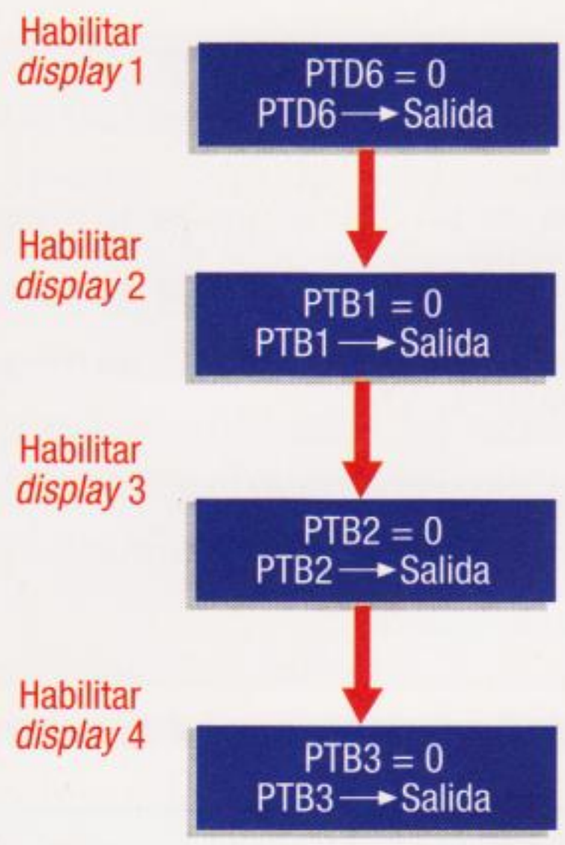


***Figura 7.8** Diagrama de flujo de la habilitación de los displays*

alto y bajo; el resultado se lleva finalmente al puerto B.

Para ejecutar el procedimiento de detectar la tecla pulsada en un momento dado, se debe enviar una señal alta por cada uno de los pines de las filas y leer el dato resultante en el puerto D.

Por ejemplo, en el momento en que se activa la fila 1 (con un cero), estamos detectando si se ha pulsado una tecla entre el 0 y el 3.

Si ha pulsado la tecla número cero "0", el puerto D debe leer el número 00111000, de manera que si se compara la lectura con ese número y son iguales, se puede determinar que la tecla "0" fue la pulsada, **figura 7.11**.

Al activar una fila se debe comparar el dato leído con todas las posibilidades de cada una de las teclas, por lo tanto los datos para establecer la comparación con cada una de las teclas se almacenan en una tabla, **figura 7.12**.

La comparación con los datos de la tabla se da para cada una de las filas. De manera que cuando se lee el mismo número 00111000 pero se está verificando la segun-

```
UpdateDisp1
        lda PORTB               ; Cargo el acumulador con el dato
        and #$0F                ; que se va a mandar por el puerto B
        sta PORTB               ; lo enmascaro y lo llevo de nuevo al puerto.
        lda datoDis1            ; Cargo el dato del primer display
        nsa                     ; intercambio de posicion los nibbles
        ora PORTB               ; alto y bajo y lo mando al puerto B.
        sta PORTB               ; Esto se debe a que se utilizan los
        jsr EnableDisp1         ; bits más altos del puerto B
        bclr UPDATE1,dispStatus ; para mandar los datos
        bset UPDATE2,dispStatus ; al display. Luego
        RTS                     ; habilito el disp1 y se muestra el dato.
UpdateDisp2
        lda PORTB               ; Ahora se repite el mismo proceso
        and #$0F                ; para el dato del display 2.
        sta PORTB               ; Se carga el dato correspondiente
        lda datoDis2            ; al segundo display, se le intercambian
        nsa                     ; de posición los nibbles alto y bajo
        ora PORTB               ; para que salgan por los pines adecuados
        sta PORTB               ; del puerto B y finalmente se habilita el
        jsr EnableDisp2         ; display dos para mostrar el resultado.
        bclr UPDATE2,dispStatus ; Estas dos últimas son banderas
        bset UPDATE3,dispStatus ; que indican que se han actualizado.
        RTS
UpdateDisp3
        lda PORTB               ; El procedimiento se repite para los displays
        and #$0F                ; tres y cuatro. De manera que con la invocación de
        sta PORTB               ; estas subsutinas secuencialmente, se actualiza
        lda datoDis3            ; y habilitan cada uno de los displays.
        nsa                     ; Intercambio de nibbles.
        ora PORTB               ; Operación lógica OR con el puerto B.
        sta PORTB               ; El resultado se queda en el puerto B.
        jsr EnableDisp3         ; Se habilita el display tres.
        bclr UPDATE3,dispStatus ; se activan las banderas
        bset UPDATE4,dispStatus ; respectivas.
        RTS                     ; Se sale de la subrutina.
UpdateDisp4
        lda PORTB               ; Este es la subrutina de actualización
        and #$0F                ; para el ultimo display.
        sta PORTB               ; De nuevo se carga el dato del registro
        lda datoDis4            ;  datoDis4, se enmascara, se intercambian
        nsa                     ; los nibbles y el resultado acondicionado
        ora PORTB               ; se retorna al puerto B.
        sta PORTB               ; Luego se habilita este display y
        jsr EnableDisp4         ; se activan las banderas respectivas
        bclr UPDATE4,dispStatus ; del registro dispStatus.
        bset UPDATE1,dispStatus
        RTS
```

***Figura 7.9** Código de la visualización de los datos en los displays*

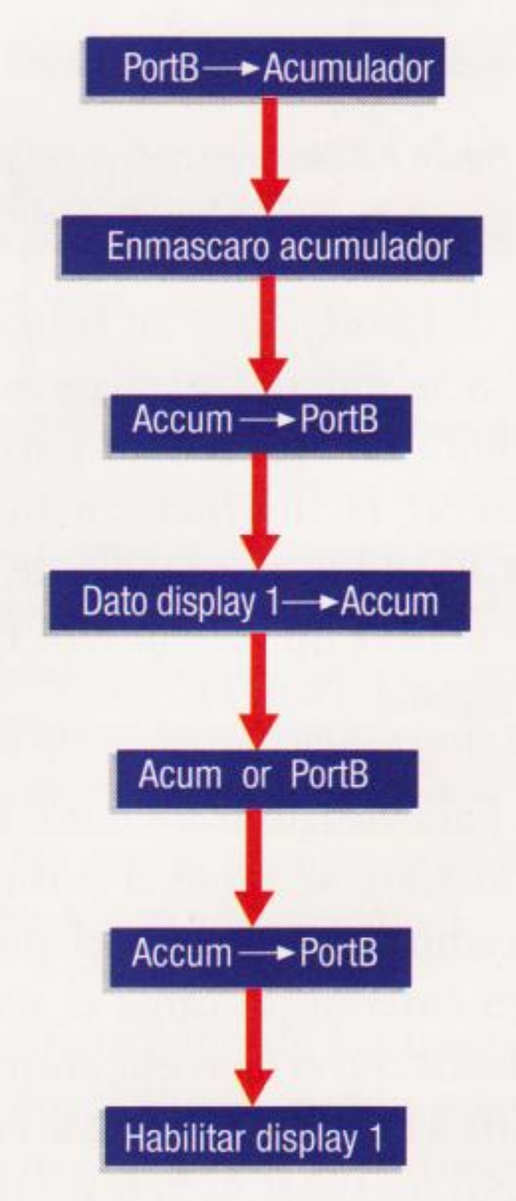


***Figura 7.10** Diagrama de flujo de la visualización de los datos en los displays*

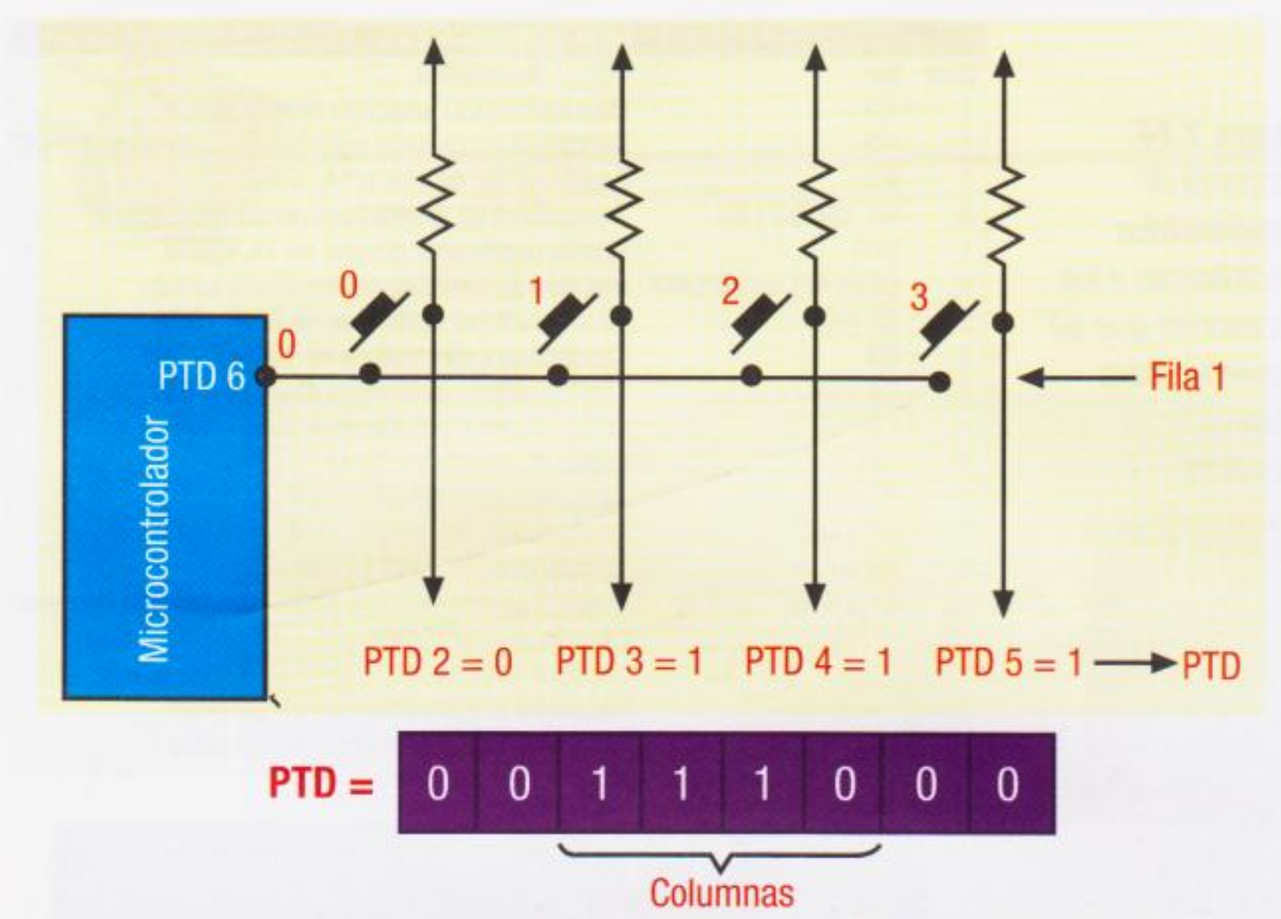


Figura 7.11 *Dato leído al pulsar la tecla "0"*

```
TECLA0   equ %00111000  ;valor del puerto para validar la tecla 0
TECLA1   equ %00110100  ;valor del puerto para validar la tecla 1
TECLA2   equ %00101100  ;valor del puerto para validar la tecla 2
TECLA3   equ %00011100  ;valor del puerto para validar la tecla 3
TECLA4   equ %00111000  ;valor del puerto para validar la tecla 4
TECLA5   equ %00110100  ;valor del puerto para validar la tecla 5
TECLA6   equ %00101100  ;valor del puerto para validar la tecla 6
TECLA7   equ %00011100  ;valor del puerto para validar la tecla 7
TECLA8   equ %00111000  ;valor del puerto para validar la tecla 8
TECLA9   equ %00110100  ;valor del puerto para validar la tecla 9
TECLAA   equ %00101100  ;valor del puerto para validar la tecla A
TECLAB   equ %00011100  ;valor del puerto para validar la tecla B
TECLAC   equ %00111000  ;valor del puerto para validar la tecla C
TECLAD   equ %00110100  ;valor del puerto para validar la tecla D
TECLAE   equ %00101100  ;valor del puerto para validar la tecla E
TECLAF   equ %00011100  ;valor del puerto para validar la tecla F
CLEAR    equ $0F        ;valor del puerto para validar la tecla clear (F)
ENTER    equ $0E        ;valor del puerto para validar la tecla enter (E)
```

Figura 7.12 *Tabla que permite establecer qué tecla fue la que se pulsó*

da fila del teclado, se debe saber que es un "4" y no un "0".

En la **figura 7.13** se muestra el código que hace la verificación de los posibles valores de la tecla oprimida para cada una de las filas. Se hace una comparación con los números establecidos en la tabla. Debido al poco espacio disponible, solo se ha colocado la lectura de la primera fila.

De acuerdo a la tecla oprimida se realiza la acción correspondiente. Si es la tecla de borrar, simplemente se borran los registros donde se almacenan los datos de los *displays*. Si es un número, esos registros se actualizan y si es un *enter* se establece el valor final de cada uno de los dígitos.

Como siempre, recuerde que el código fuente completo está disponible gratuitamente en nuestra página en la internet:

www.cekit.com.co

en el enlace de este curso.

```
LeaTeclaFila1
        lda PORTD          ; Cargo en el acumulador
        and #%00111100     ; el dato del puerto D.
        cmp #NO_TECLA      ; Y empiezo a comparar con
        beq noKey1         ; los datos de la tabla.
        cmp #TECLA0        ; Si al comparar con el
        beq isTecla0       ; dato que indica que no
        cmp #TECLA1        ; hay ninguna tecla oprimida
        beq isTecla1       ; en esta fila, se sale para
        cmp #TECLA2        ; chequear la siguiente.
        beq isTecla2       ; se compara con cada número
        cmp #TECLA3        ; posible y se salta a la rutina
        beq isTecla3       ; correspondiente cuando se ha
noKey1 RTS                 ; establecido una igualdad.
isTecla0
        bset 1,wasTecla ; Esta es la primera, se trata
        bset 0,NewTecla ; de la detección de la
                        ; tecla 0.
        mov #0,tecla    ; Se activan las banderas
                        ; respectivas
        RTS             ; y se carga el registro tecla
                        ; con 0.
isTecla1
        bset 1,wasTecla ; Esta es la rutina cuando
                        ; se detecta
        bset 0,NewTecla ; la tecla 1. De nuevo se
                        ; activan las
        mov #1,tecla    ; banderas y se carga
                        ; tecla con 1.
        RTS
isTecla2
        bset 1,wasTecla ; Las banderas que se
                        ; activan sirven
        bset 0,NewTecla ; para indicar que se ha
                        ; detectado
        mov #2,tecla    ; una tecla oprimida. Es
                        ; para señalizacion
        RTS             ; en otras partes del
                        ; programa.
isTecla3
        bset 1,wasTecla ; Esta es la última
                        ; subrutina en la cual
        bset 0,NewTecla ; se detecta la pulsación
                        ; de la tecla 3.
        mov #3,tecla    ; Se carga tecla con el
                        ; número 3.
        RTS
```

Figura 7.13 *Código de lectura y detección de la tecla oprimida*

Ensamblaje

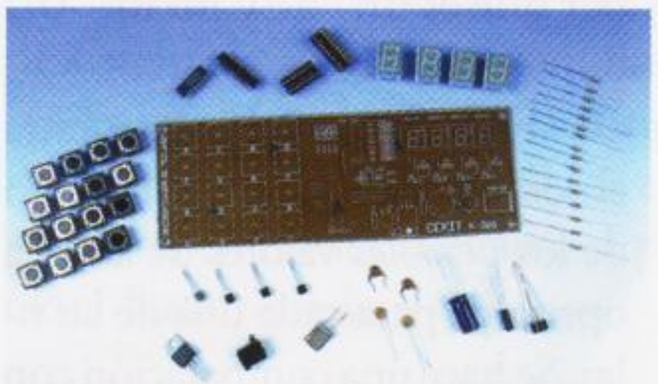
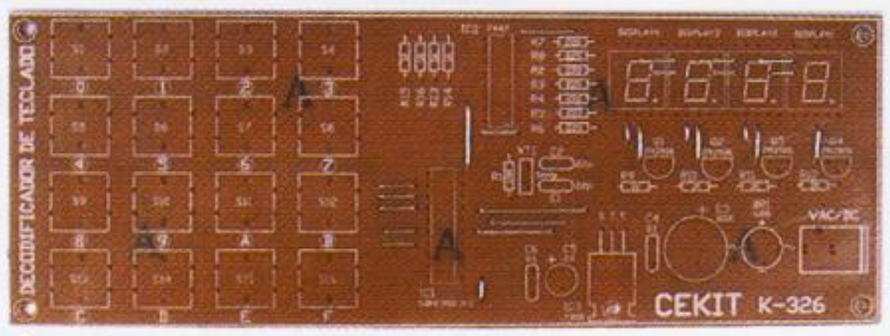
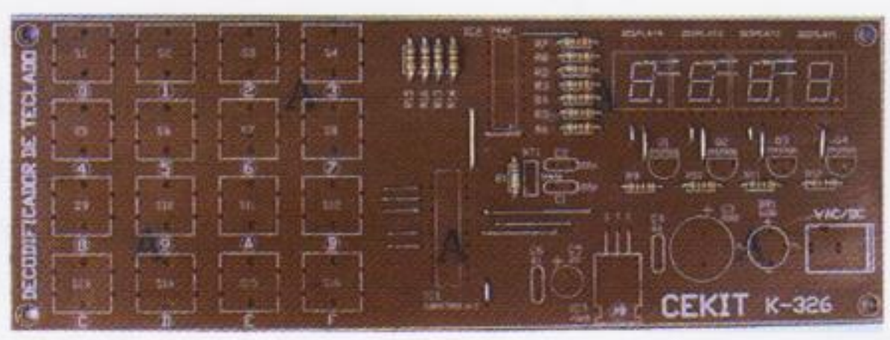


Figura 7.14
Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el proyecto.

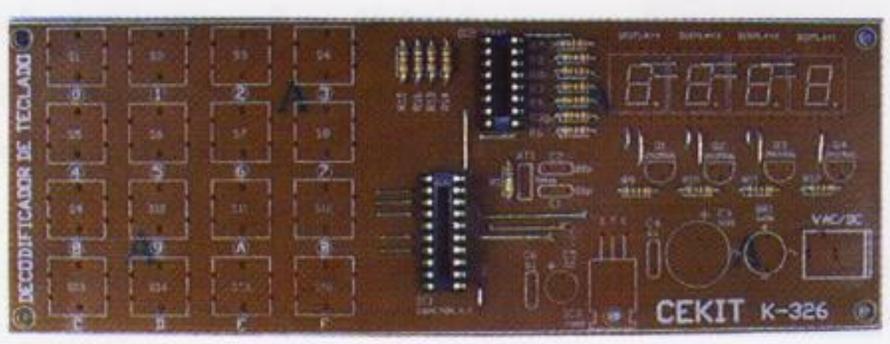
LISTA DE MATERIALES		
Cant.	**Ref.**	**Descripción**
1	IC1	Microcontrolador motorola 68HC908JK3
1	IC2	Decodificador de siete segmentos ánodo común, 7447
1	IC3	Regulador fijo de +5V a 1A, 7805
4	Q1, Q2, Q3 y Q4	Transistores de propósito general PNP, 2N3906
1	BR1	Puente rectificador circular de 1A, W06M
4	DISPLAY1 a DISPLAY4	Displays de siete segmentos ánodo común
2	C1 y C2	Condensadores cerámicos de 22pF - 50V
1	C3	Condensador electrolítico de 1000uF - 25V
2	C4 y C6	Condensadores cerámicos de 0.1uF - 50V
1	C5	Condensador electrolítico de 22uF - 16V
1	XT1	Cristal de quarzo de 5MHz
1	R1	Resistencia de 10MΩ - 1/4W
7	R2 a R8	Resistencias de 220Ω - 1/4W
8	R9 a R16	Resistencias de 1KΩ - 1/4W
16	S1 a S16	Pulsadores grandes de 4 pines, para circuito impreso
1	K-326	Circuito impreso CEKIT
1		Conector CA/CC
1		Base para circuito integrado de 20 pines
1		Base para circuito integrado de 16 pines



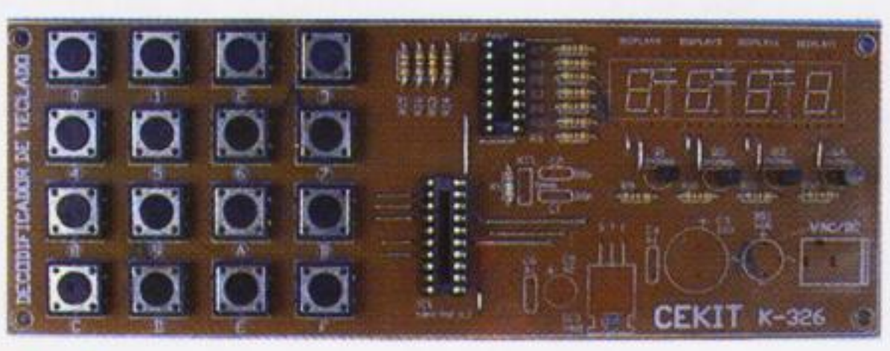
Paso 1
Con un alambre delgado, instale y suelde los puentes marcados en el circuito impreso. ***Figura 7.15***



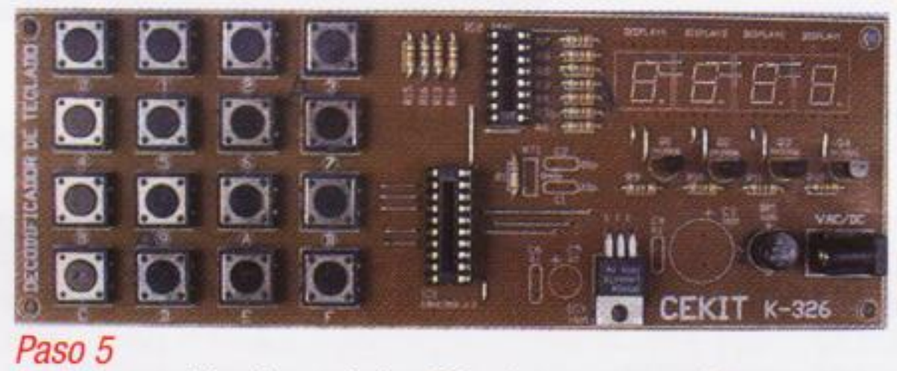
Paso 2
Luego instale y suelde las resistencias (16).
Figura 7.16



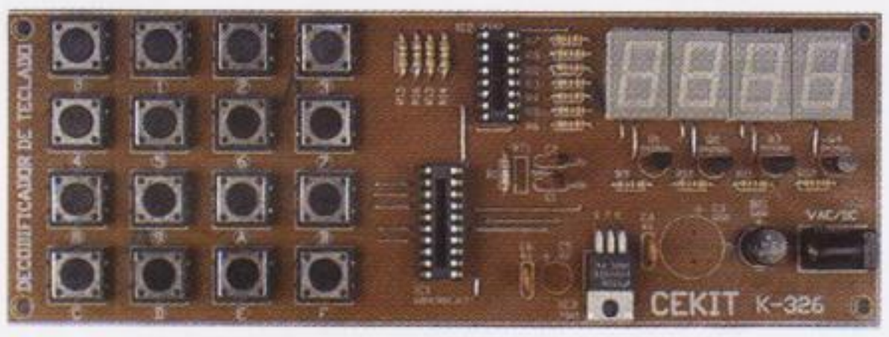
Paso 3
Instale y suelde las bases o sockets para los dos circuitos integrados. ***Figura 7.17***



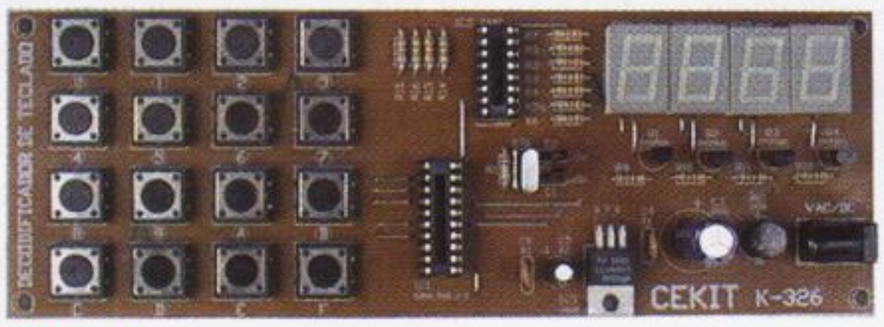
Paso 4
Instale y suelde los cuatro transistores PNP y las teclas que forman el teclado hexadecimal (16), observando su posición correcta. ***Figura 7.18***



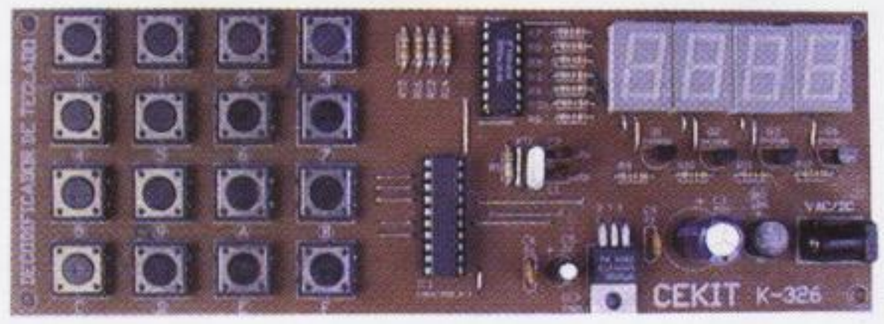
Paso 5
Instale y suelde el regulador IC3, el puente rectificador BR1 y el conector de alimentación CA/CC. ***Figura 7.19***



Paso 6
Instale los cuatro displays y los cuatro condensadores cerámicos C1, C2, C4 y C6. ***Figura 7.20***

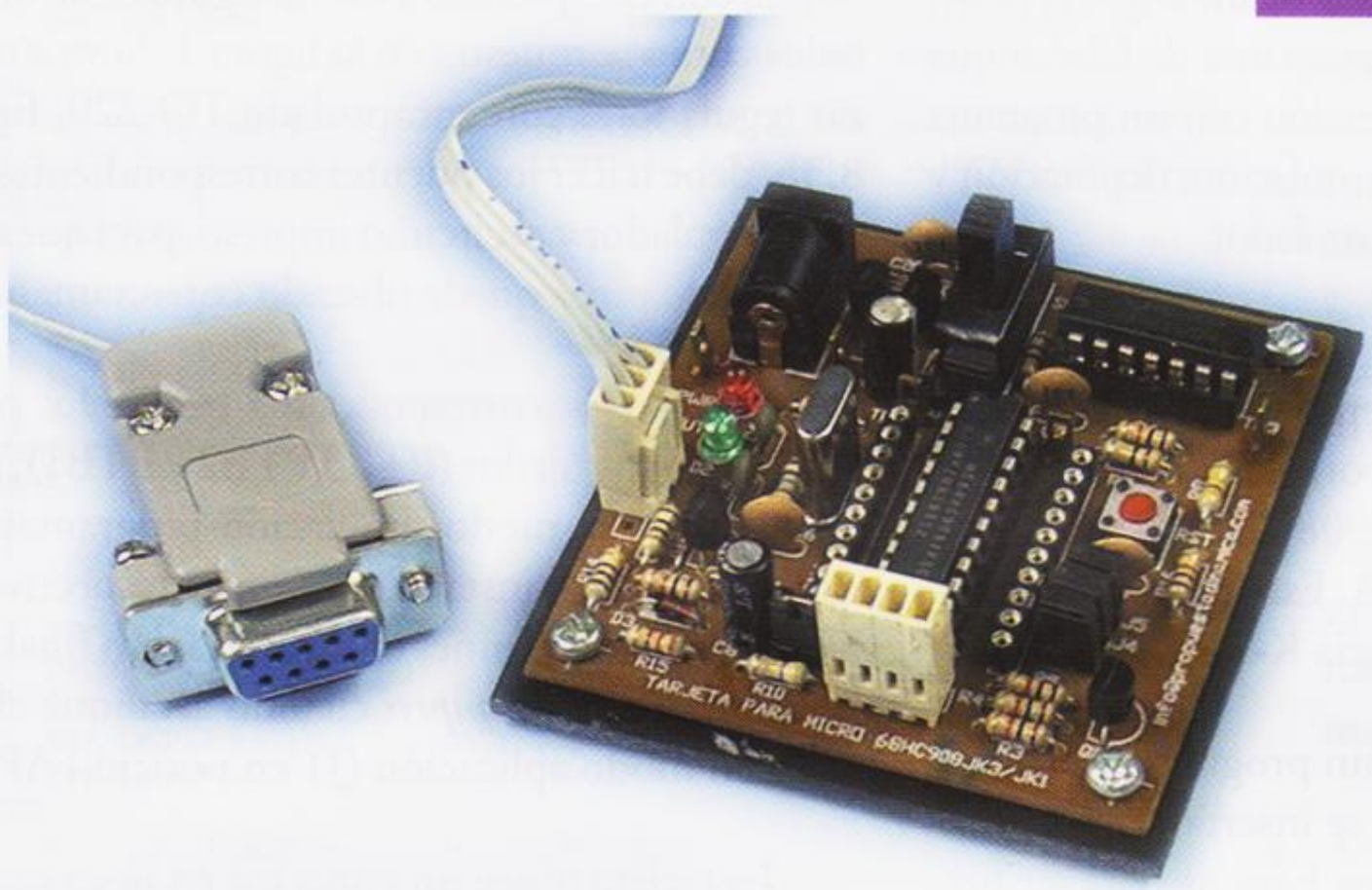


Paso 7
Instale el cristal XT1 y los condensadores electrolíticos C3 y C5 observando su posición ***Figura 7.21***



Paso 8. Prueba final
Instale los circuitos integrados. Alimente el circuito con una fuente regulada o no, hasta de 9 voltios. Los displays se deben activar indicando el número 0000. De no ser así, asegúrese de que los integrados estén bien ubicados y que los niveles de voltaje estén correctos. Pulse alguna tecla y observe su visualización.

PROYECTO 8



SISTEMA DE DESARROLLO PARA MICROCONTROLADORES MOTOROLA MC68HC908JK3/JK1

Esta tarjeta de desarrollo para microcontroladores Motorola permite editar, ensamblar, simular y programar los microcontroladores 68HC908JK3, y así construir proyectos de manera sencilla, rápida y económica.

El MC68HC908JK3 es uno de los miembros de la familia 08 de Motorola, caracterizados por su bajo costo y alto desempeño. Dentro de las principales características de estos microcontroladores tenemos su memoria *Flash* (borrable y programable eléctricamente), diez canales de conversión de analógico a digital (ADC) y, tal vez lo más importante, un bloque de memoria denominado ROM MONITOR, en el cual se almacena un programa de fábrica que permite establecer comunicación con un programa de computadora y hacer simulación, depuración y programación del microcontrolador.

Funcionamiento

El proyecto que construiremos consiste en un sistema de desarrollo que soporta los microcontroladores JK3 y el MC68HC908JK1 (JK1) de la familia 08 de MOTOROLA. Este proyecto se puede adquirir bajo la referencia K-218 de CEKIT.

El sistema consta de un programa (*software*) y de una tarjeta en la cual se inserta el microcontrolador con los elementos básicos para su funcionamiento; el diagrama esquemático de esta tarjeta se puede apreciar en la **figura 8.1**. En la **figura 8.2** podemos apreciar una guía de ensamblaje que nos servirá como referencia al instalar los elementos. La tarjeta posee entrada para alimentación de CC, conversores de nivel TTL a RS-232 que permiten conexión serial a una computadora para simulación y programación, circuito oscilador basado en cristal y un LED indicador de alimentación.

El *jumper* J1 selecciona el modo de funcionamiento. En la posición «PROG» el sistema arranca en modo SIMULACIÓN-PROGRAMACIÓN, permitiendo mediante el programa instalado, hacer simulación y posterior programación del *chip*. J1 en la posición «APP» permite que el sistema arranque en modo aplicación; es decir, ejecute el programa grabado por el usuario.

SW1 es un interruptor tipo corredera de dos polos y dos posiciones, que permite remover la alimentación completa del sistema cuando el programa de la PC así lo solicite para validar el código de seguridad interno del microcontrolador. El pulsador RST está conectado a la señal de *reset* del microcontrolador permitiendo el control por el usuario de esa función.

En el circuito de alimentación, se deben instalar preferiblemente reguladores tipo 78L05 y 78L08 con encapsulado TO-92, **figura 8.3a**, orientados como se muestra en la figura 1. Si va a utilizar reguladores con encapsulado TO-220, **figura 8.3b**, debe hacer los puentes correspondientes entre el regulador y el circuito impreso, para que cada uno de los pines quede ubicado correctamente.

J3, J4 y J5 corresponden a tres de los pines del microcontrolador (PTB3 PTB2 y PTB1); para poder entrar al modo simulación-programación, ellos deben tener instalado el *jumper* respectivo. Si usted utiliza estos pines en su aplicación final, deberá remover los *jumpers* cuando arranque el sistema en modo aplicación (J1 en posición APP).

La tarjeta posee un conector en línea macho de 3 pines (CN2), el cual permite la conexión directa a la PC (para simulación-programación) y mediante los conectores en línea CN3, CN4 y CN5 se tiene acceso a todos los pines del *chip*, con los cuales puede conectar el sistema de desarrollo a su aplicación final.

Adicionalmente, el sistema cuenta con un LED verde conectado a uno de los pines del microcontrolador (PD7) y un *jumper* que conmuta entre +VDD y tierra conectado al pin PB5; mediante estos dos elementos ilustraremos el manejo de entradas y salidas.

En el momento del ensamblaje, tenga la precaución de soldar primero los componentes de bajo nivel, tales como resistencias y condensadores; después puede continuar con los transistores y demás elementos. Al finalizar, remueva los excesos del fundente de la soldadura con algún limpiador electrónico de contactos o con *thinner*, y construya el cable de interfaz con la computadora, haciendo los puentes como se ilustra en la **figura 8.4** (en el conector DB9: el pin 4 unido con el pin 6 y el pin 7 unido con el pin 8).

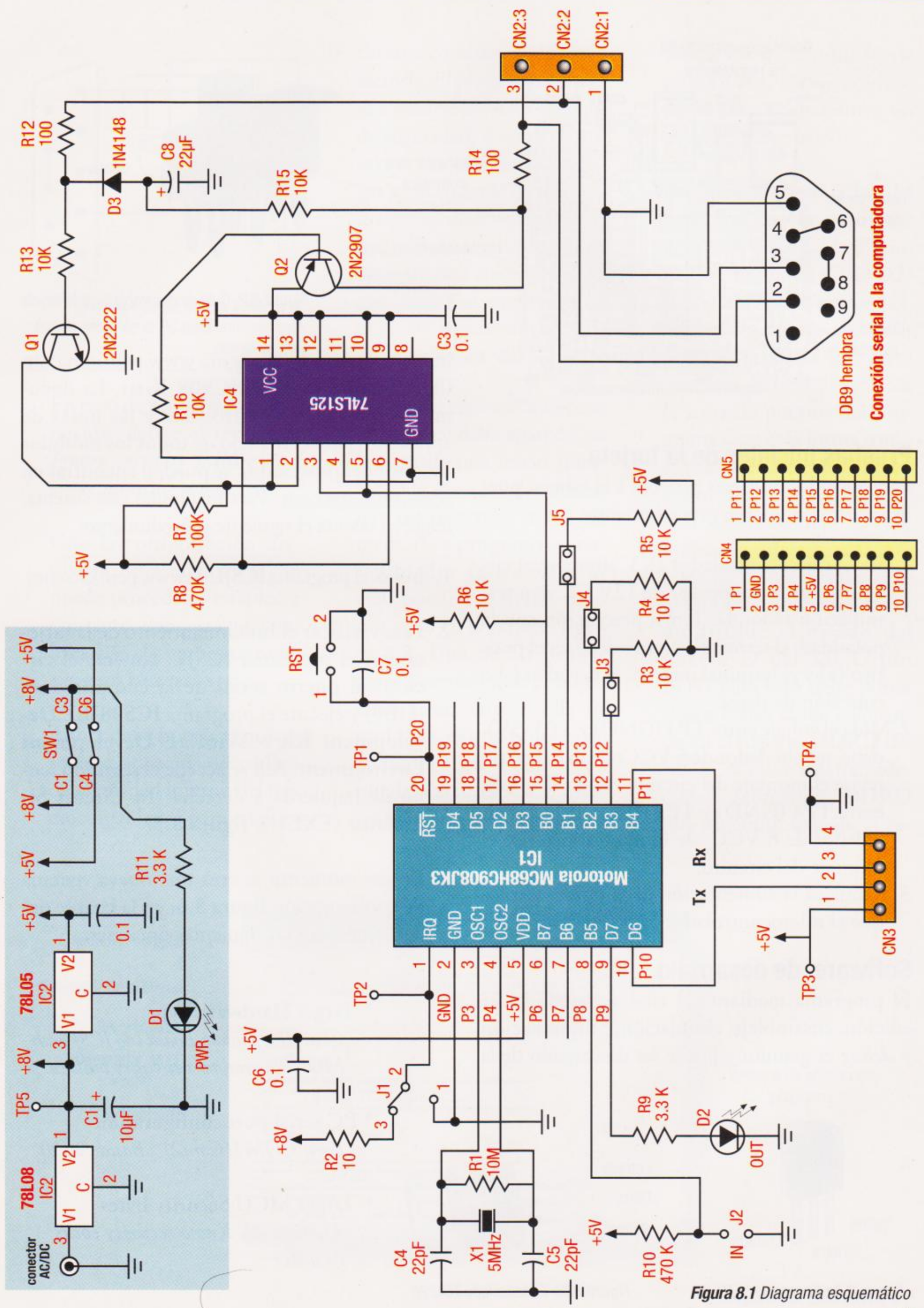


Figura 8.1 Diagrama esquemático

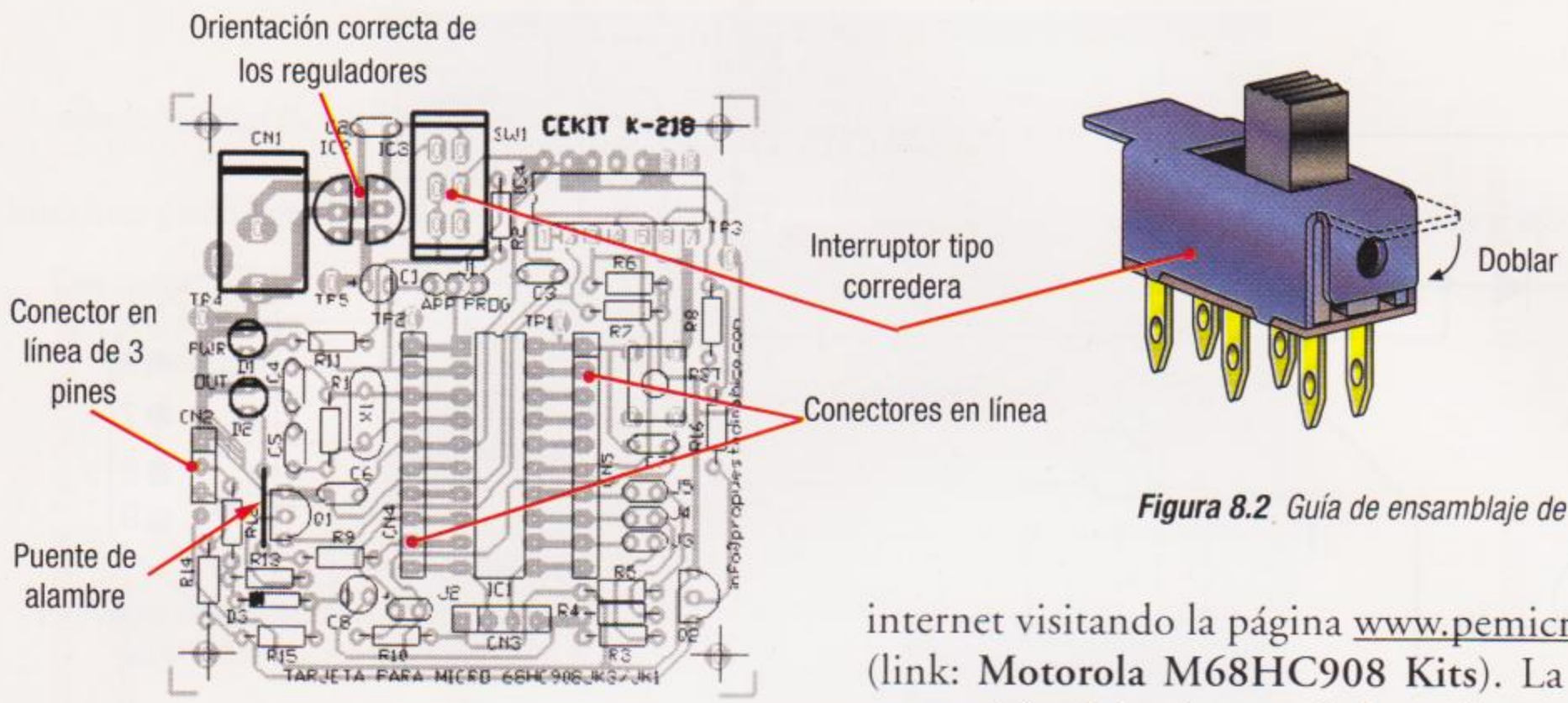


Figura 8.2 *Guía de ensamblaje de la tarjeta*

Pruebas iniciales de la tarjeta

Para hacer las primeras pruebas a la tarjeta, puede seguir el siguiente procedimiento:

1. Sin el microcontrolador JK3, alimente el circuito con un adaptador de 12V CC con terminación redonda. Tenga precaución con la polaridad: el terminal externo debe ser el positivo (+) y el terminal interno, el negativo (-) o conexión de tierra.
2. Mida el voltaje entre **TP4** (GND) y **TP3** (5V), debe ser alrededor de 5 VCC; de lo contrario revise el montaje del circuito. Mida el voltaje entre **TP4** (GND) y **TP2** (IRQ), debe ser alrededor de 8 VCC; de lo contrario revise el montaje del circuito.
3. Remueva la alimentación de la tarjeta y coloque el microcontrolador **JK3** en la base.

Software de desarrollo

El programa mediante el cual se puede hacer edición, ensamblaje, simulación, programación y *debug* es gratuito y puede ser descargado de la internet visitando la página www.pemicro.com (link: **Motorola M68HC908 Kits**). La documentación del microntrolador y las notas de aplicación para el manejo de todos los módulos internos del procesador, se pueden encontrar en www.mot-sps.com. Para manipular este sistema, tenga en cuenta el siguiente procedimiento:

1. Instale el programa **ICSJL** de www.pemicro.com.
2. Para verificar el funcionamiento de la tarjeta con el programa **ICSJL**, conecte el circuito al puerto serial de la computadora (DB9) y ejecute el programa **ICS08JLZ Development Kit > WinIDE Development Environment**. Allí se accede el segundo icono de izquierda a derecha (**In-Circuit Simulator (EXE1)**), **figura 8.5**.
3. En ese momento se crea una nueva ventana de configuración **figura 8.6**, en la cual se deben establecer las siguientes opciones:

   * **Target Hardware Type**
     · *Class III - Vustom Board (no ICS) with MON08 serial port circuitry built in.*

   * **PC serial port configuration**
     · *Port: COM1(Com2) Baud: 9600*

   * **Target MCU Security bytes**
     · *Attempt All Know security codes in order*

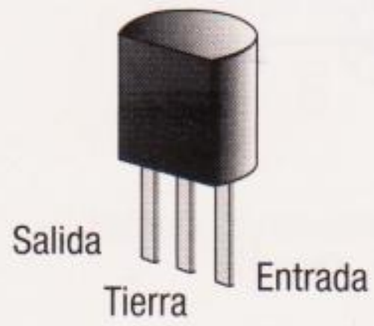


Figura 8.3a *Encapsulado TO-92*

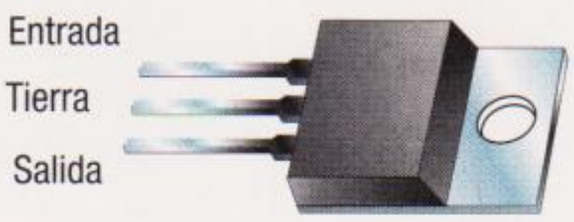


Figura 8.3b *Encapsulado TO-220*

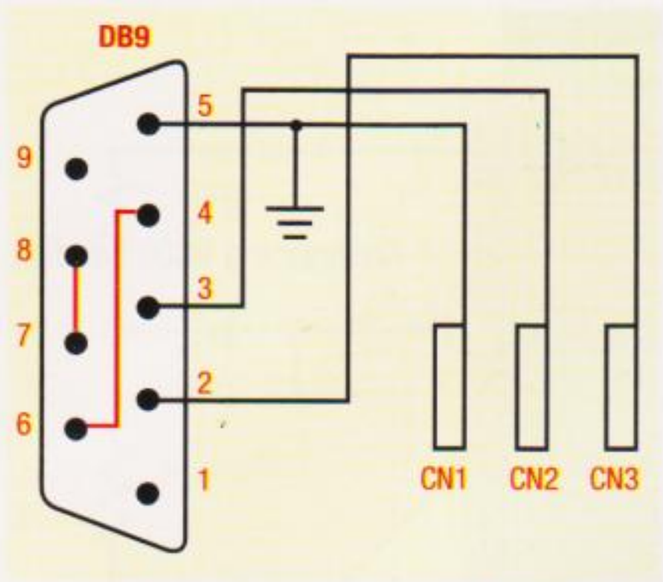


Figura 8.4 *Cable que se conecta al puerto serial de la PC para comunicarse con la tarjeta*

En caso de problemas de conexión, active el recuadro: ***Ignore security failure and enter monitor mode.***

Con la configuración dispuesta de esta manera, se puede proceder a establecer contacto con esta tarjeta, haciendo clic en ***Contact target with this settings...***

Ahora debe iniciar el proceso de verificación del código de seguridad interno del microcontrolador para establecer una comunicación satisfactoria. El programa empieza a ensayar diversos códigos de seguridad, y antes de ensayar cada uno, se debe apagar y encender la tarjeta mediante el interruptor SW1, para lo cual el programa muestra una nueva ventana que indica el momento en que esta operación se debe realizar, **figura 8.7**.

4. Al obtener comunicación con el circuito, debe aparecer en la pantalla una nueva interfaz en la que se tienen ventanas de puertos, zonas de memoria y programa en ensamblador (*assembler*), el cual puede ejecutarse paso por paso en la tarjeta, **figura 8.8**. Una de las principales herramientas con que cuenta este entorno de simulación es una ventana de comandos, en la cual usted puede dar órdenes e interactuar con el programa a través de un conjunto de comandos establecidos, cuya lista se encuentra en el menú de ayuda.

5. Para ilustrar la evaluación del *hardware* con la ayuda del *software* **ICSJLZ**, el circuito provee una salida (LED **D2**) y una entrada (*Jumper* **J2**). Coloque y remueva el *jumper* **J2** de la tarjeta. Este cambio debe verse reflejado en la zona de puertos del programa (esta es la forma como se puede evaluar que las entradas de su proyecto funcionen adecuadamente). Para evaluar la salida acceda a la línea de comandos del programa (inferior izquierda) y modifique el estado del puerto D del MCU pin PD7; para esto debe poner el pin como salida colocando en el registro DDRD del MCU el valor $80 (hexadecimal), (comando: **DDRD 80**) y a continuación, el va-

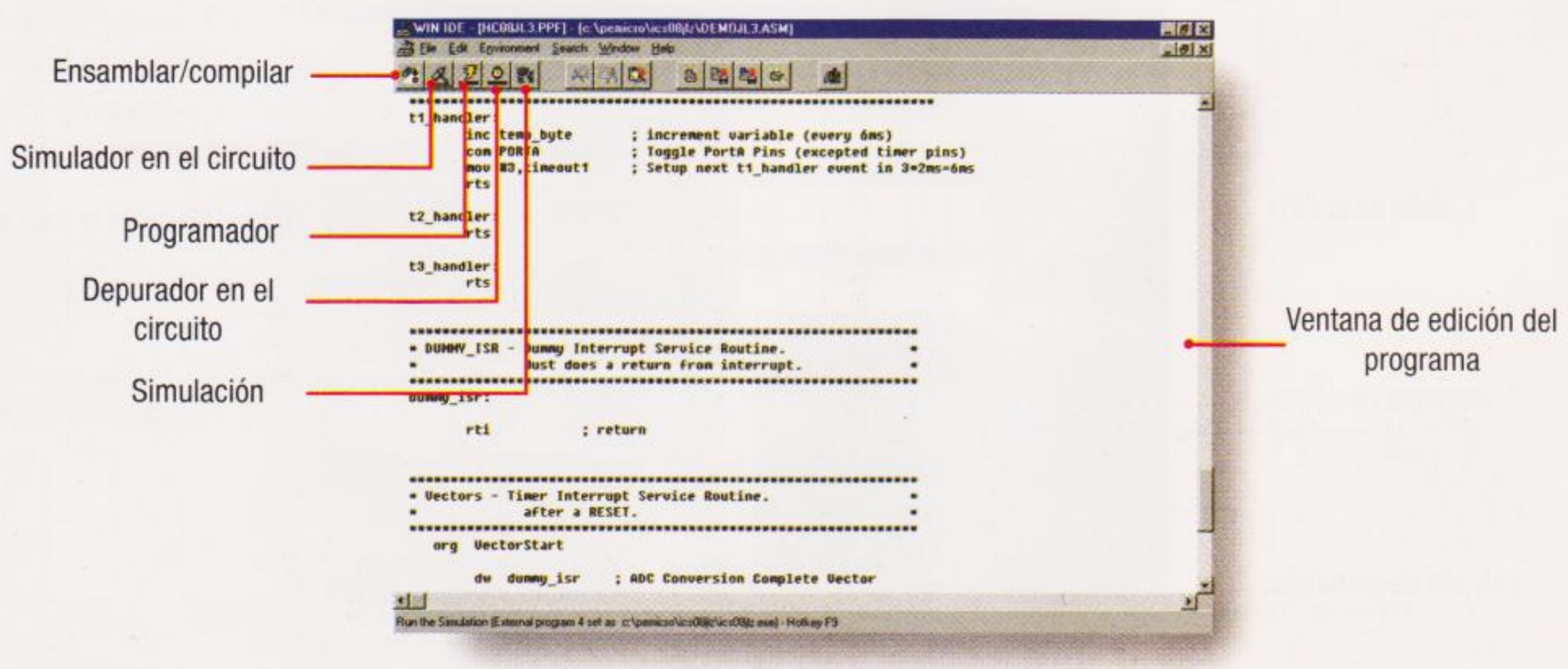


Figura 8.5 *Entorno Win IDE*

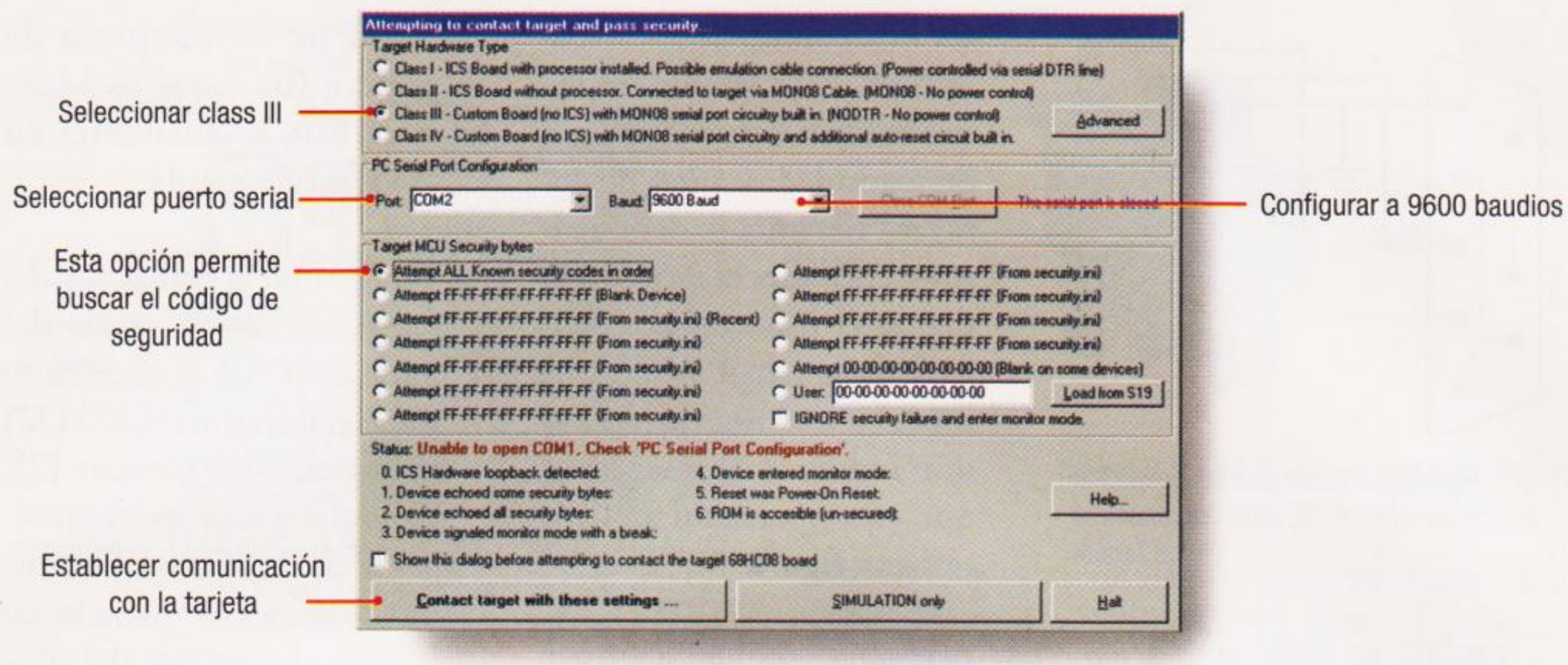


Figura 8.6 *Entrada al programa simulador del sistema y configuración de la tarjeta*

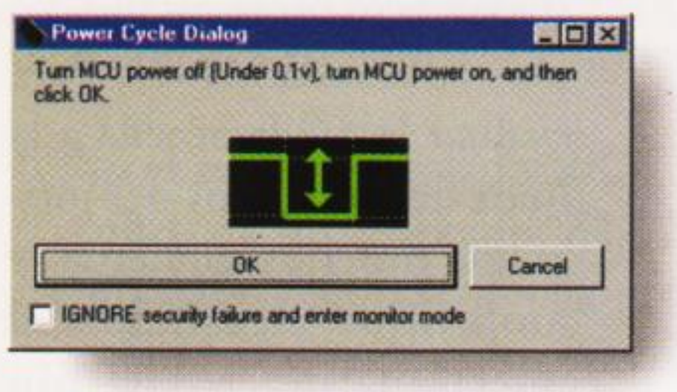


Figura 8.7 *Ventana de señalización de encendido y apagado de la tarjeta*

lor del pin en el registro PORTD del MCU en el estado requerido (para nuestro caso en "1" lógico), (comando: PORTD 80). Después de la ejecución de estos dos comandos, el LED **D2** debe encenderse, indicando la conexión adecuada (de manera similar se pueden evaluar las demás salidas del proyecto en particular).

6. La simulación de programas puede hacerla sin la conexión del circuito, accediendo al botón **SIMULATION only** (esta simulación es recomendable para zonas de programa que no tengan interacción directa con el *hardware*).

Programa ejemplo

El ejemplo que ilustramos a continuación consiste en un sencillo programa que varía la frecuencia de oscilación de encendido del LED D2, dependiendo del estado de la entra-

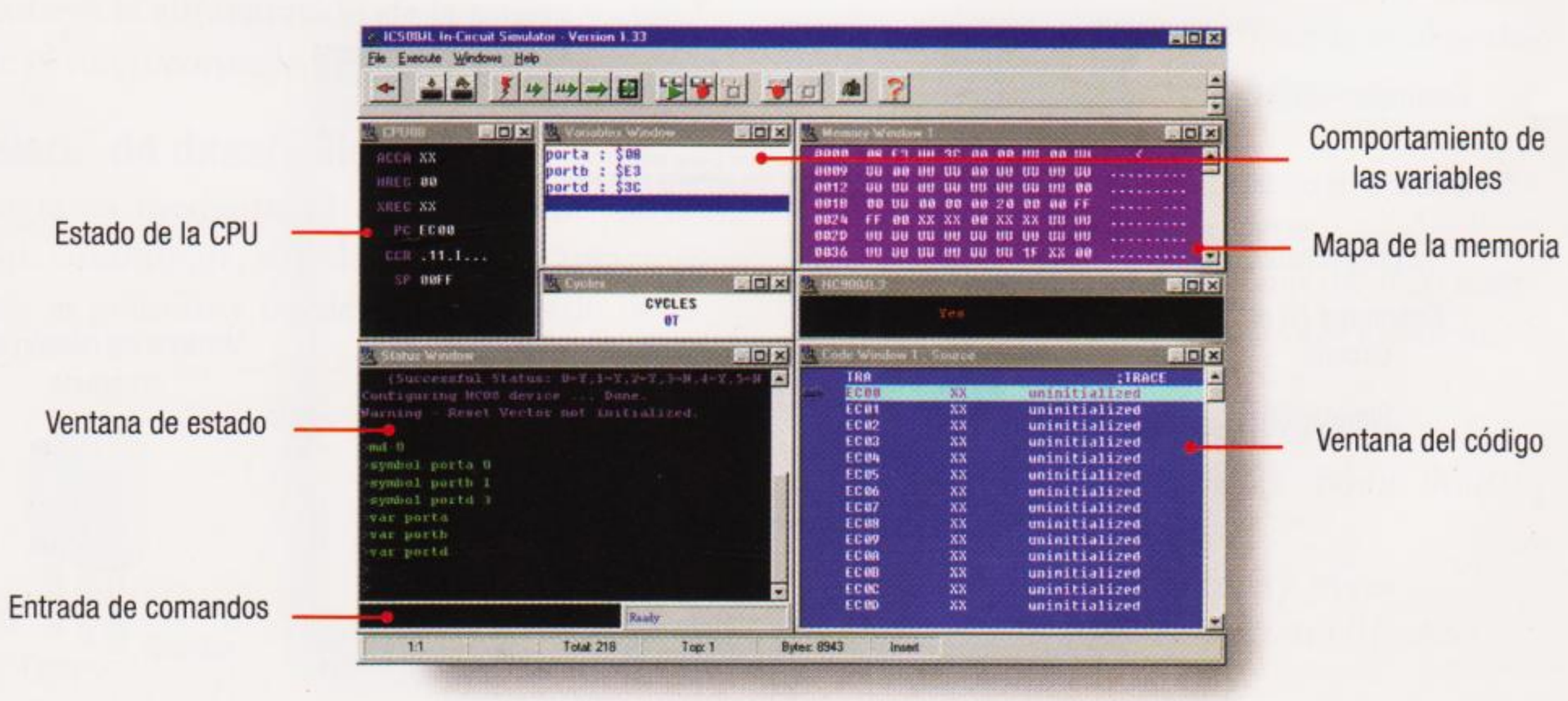


Figura 8.8 *Entorno del programa de simulación*

```
$Include 'jl3regs.inc'              ;Archivo de definición de registros
FLASH_START_JK3       equ $EC00     ;dirección de inicio de la flash
RESET_VEC             equ $FFFE     ;dirección del vector de reset
;definición de bits
COPD          equ 0                 ;bit de CONFIG1 para habilitar/
                                    ;deshabilitar el watchdog
PD7           equ 7                 ;bit de conexión a la salida (Led D2)
PB5           equ 5                 ;bit de conexión de la entrada (jumper J2)
org FLASH_START_JK3
Start:
      Rsp                           ;inicializa stack pointer
      Bset COPD,CONFIG1             ;deshabilita el COP (watchdog)
      Bsr Init_Led                  ;realiza llamado a la subrutina de
                                    ;estado inicial del led
LoopD:
Brset PB5,PORTB,Half1               ;verifica el estado de la entrada J2
Lda #255T                           ;carga el AccA con el valor 255 decimal
Bra Dly1
Half1: Lda #70T                     ;J2 colocado. Se carga el AccA con un
                                    ;valor menor
Dly1:  Bsr Delay                    ;invoca subrutina de retardo
       Deca                         ;decrementa el AccA
       Bne Dly1
       Bsr Led_ON                   ;enciende Led D2
Brset PB5,PORTB,Half2               ;verifica el estado de la entrada J2
Lda #255T                           ;carga el AccA con el valor 255 decimal
Bra Dly2
Half2: Lda #70T                     ;J2 colocado. Se carga el AccA con
un                                  ;valor menor
Dly2:  Bsr Delay                    ;invoca subrutina de retardo
       Deca                         ;decrementa el AccA
       Bne Dly2
       Bsr Led_OFF                  ;apaga Led D2
       Bra LoopD
Led_OFF
       Bclr PD7,PORTD               ;PD7 en 0
       RTS
Led_ON:
       Bset PD7,PORTD               ;PD7 en 1
       RTS
;subrutina de retardo básica
Delay
       ldhx #$FFFE                  ;carga registro doble H:X con valor inicial
Loop1   Aix #-1                     ;decrementa el registro H:X
       Cpx #0                       ;compara con 0
       Bne Loop1                    ;si no ha llegado continua el retardo
       RTS                          ;retorna del llamado
Init_Led:
       Bset PD7,DDRD                ;establece el pin PD7 del MCU como salida
       Bclr PD7,PORTD               ;apaga el led D2
       RTS                          ;retorna de la subrutina
       Org RESET_VEC
       Dw Start                     ;al darse reset salta a Start
```

Figura 8.9 *Programa ejemplo*

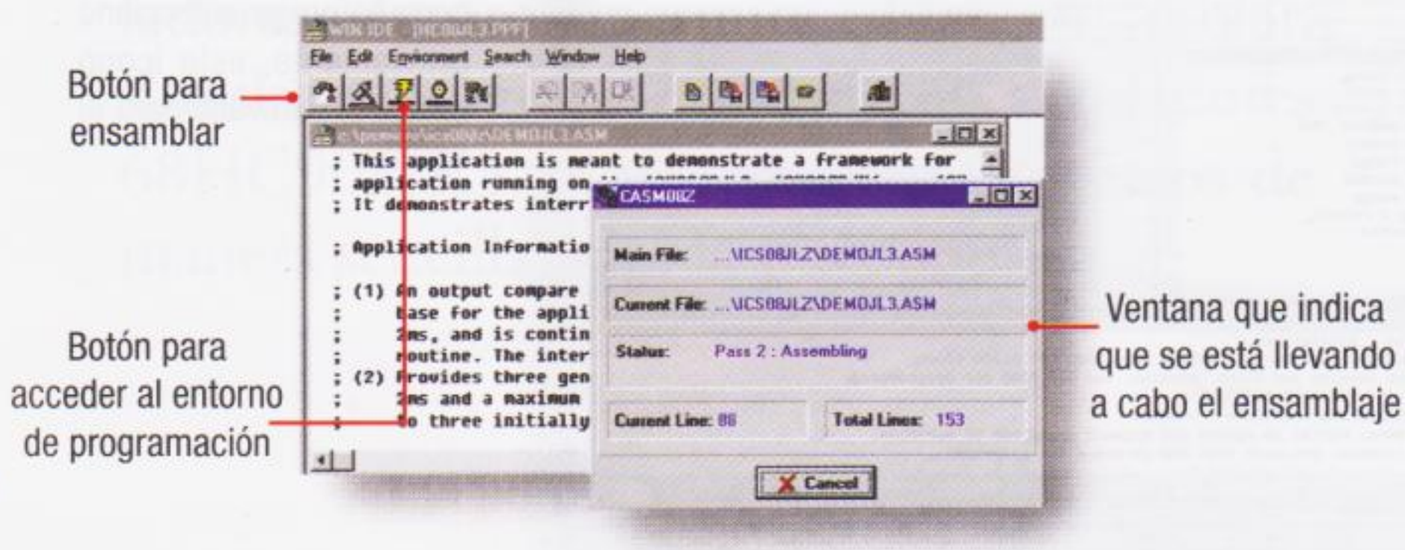


Figura 8.10 *Ensamblaje del programa*

da J2 (*jumper* puesto, menor frecuencia de oscilación), **figura 8.9**. Para la programación del MCU con el programa ejemplo, utilice los pasos que siguen a continuación.

1. Digite el programa en el editor del WinIDE.

2. Ensamble el programa haciendo clic en el primer icono (de izquierda a derecha) de la barra de herramientas, **figura 8.10**.

3. Acceda el tercer icono del programa WinIDE Development environment (**programmer** (EXE2)).

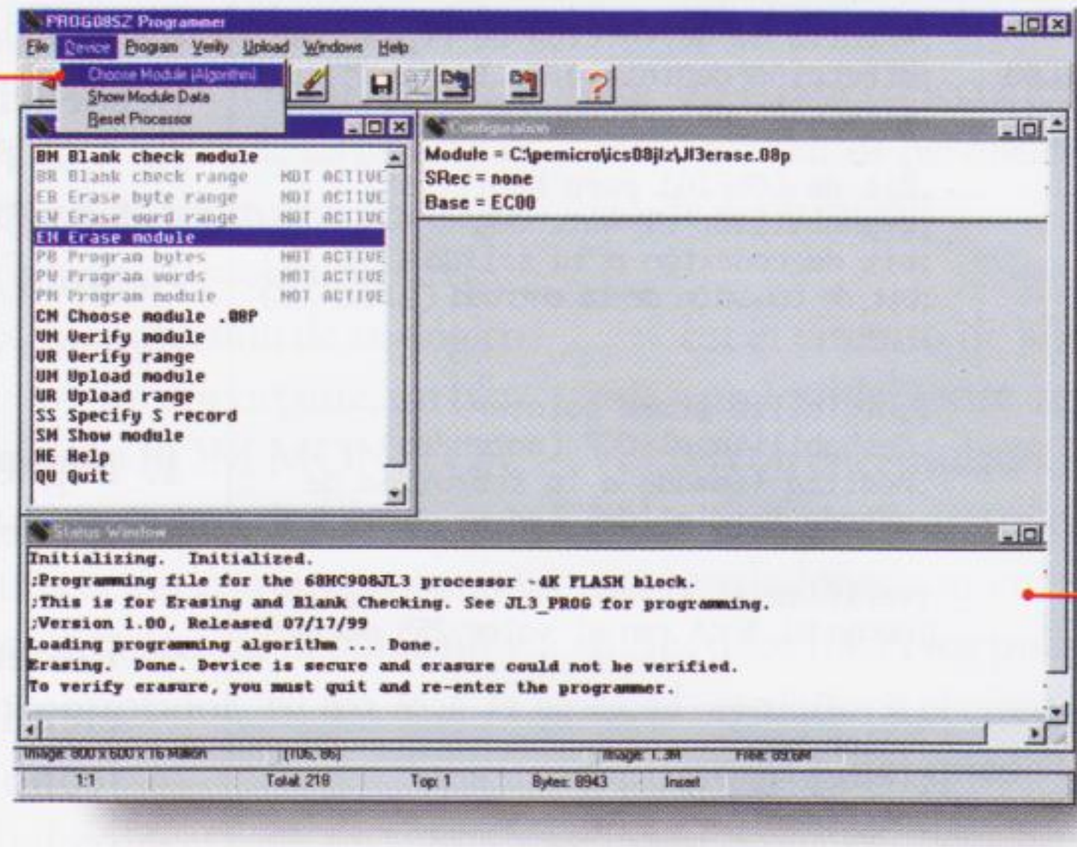
4. Antes de grabar cualquier programa en la memoria del microcontrolador, éste se debe borrar. Al entrar al *software* de programación debe escoger entre el modo borrar y el modo programar. Esto se logra haciendo clic en el menú **device>algoritm**, **figuras 8.11** y **8.12**.

5. Borrar el MCU presionando el icono 6 (de izquierda a derecha el icono con borrador sobre el *chip*), **figura 8.13**.

6. Cargue en memoria el algoritmo de programación del MCU JK3 (**908_JK3.08**), **figura 8.14**.

7. Especifique el archivo que va a programar mediante el icono 7 (de izquierda a derecha, el icono con disquete), **figura**

Antes de programar un microcontrolador Motorola, éste se debe borrar. Al entrar al entorno de programación se debe seleccionar el algoritmo que se va a utilizar siendo las opciones: borrar y programar. Haciendo clic en este menú elegimos el tipo de algoritmo.



Esta ventana de estado informa qué algoritmo se ha cargado y las acciones que se ejecutan sobre el microcontrolador.

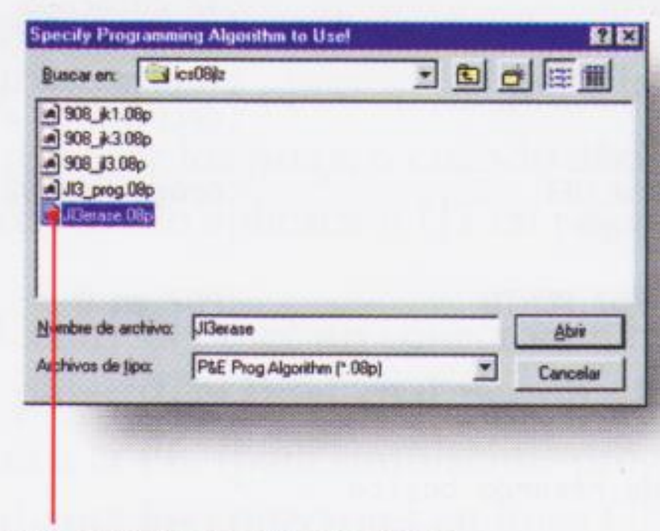
Figura 8.11 *Entorno de programación*

8.15. Los archivos binarios que se graban en el microcontrolador poseen la extensión s19.

8. Programar el *chip* presionando el icono 8 (de izquierda a derecha el icono de rayo sobre el *chip*), **figura 16**.

9. Para ensayar el programa en la tarjeta, cambie el *jumper* J1 de la posición 2-3 (**PROG**) a la posición 1-2 (**APP**); desenergice y energice la tarjeta nuevamente para proporcionar un *reset*.

10. Para volver al modo simulación *in-circuit* y programación, cambie J1 de la posición 1-2 (**APP**) a la posición 2-3 (**PROG**), remueva la alimentación y conéctela nuevamente.



Este es el algoritmo que se debe seleccionar para borrar el microcontrolador. Es el primer paso en la programación.

Figura 8.12 *Selección del algoritmo que se va a trabajar. Inicialmente el de borrado*

Haciendo clic en este icono damos inicio al proceso de borrado. Si el algoritmo de borrado no se seleccionó previamente, este icono estará desactivado para el usuario.

Este indicador nos dice que se ha cargado el algoritmo de borrado.

Al momento de borrar, en esta ventana se muestra el aumento progresivo de este procedimiento en el microcontrolador.

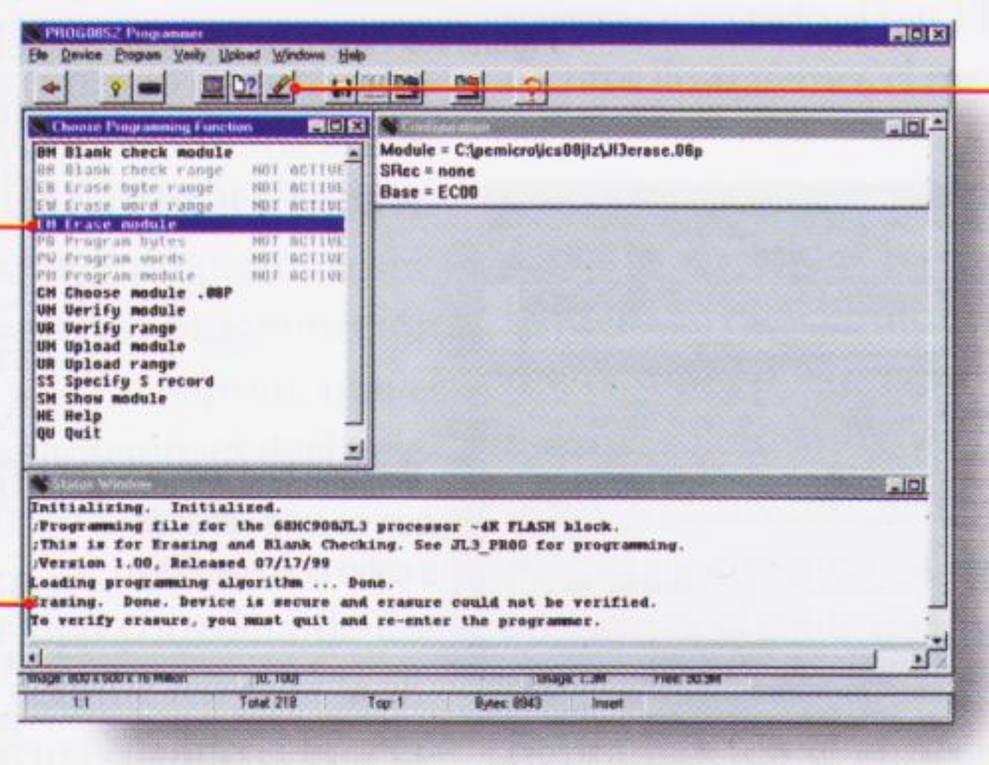


Figura 8.13 *Borrado del microcontrolador*

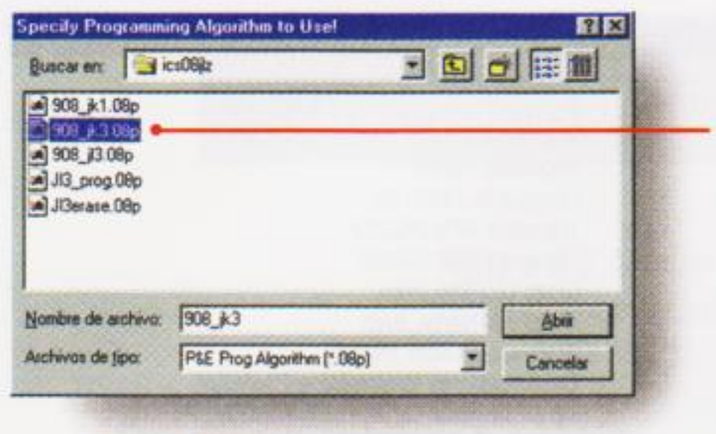


Figura 8.14 *Selección del algoritmo de programación*

Una vez borrado el microcontrolador lo puede programar. Para iniciar este proceso debe cambiar el algoritmo de borrado por el de programación seleccionando la opción que se muestra en esta figura. A ella puede acceder en el menú ***Device>>Choose algorithm***.

Haciendo clic en el icono del disquete, puede cargar el archivo binario resultante de ensamblar el código fuente, y el cual se va a grabar en la memoria *Flash* del microcontrolador. Los microcontroladores Motorola manejan un formato *.S19 para sus archivos binarios. En este momento seleccione el archivo que va a trabajar en esta ventana, el cual tiene el mismo nombre que el del código fuente.

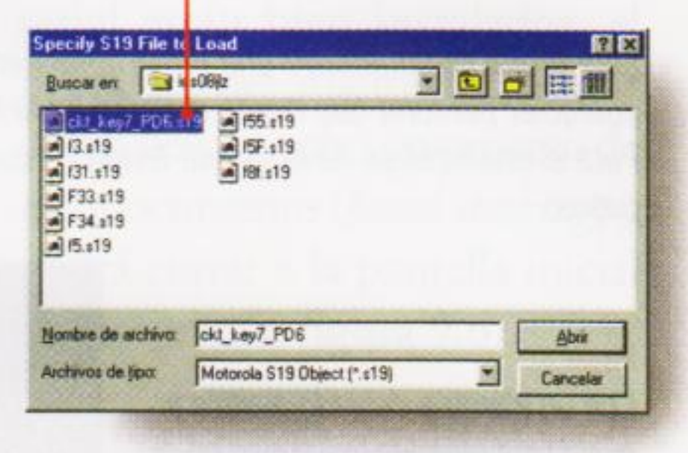


Figura 8.15 *Selección del archivo binario que se va a grabar en el microcontrolador*

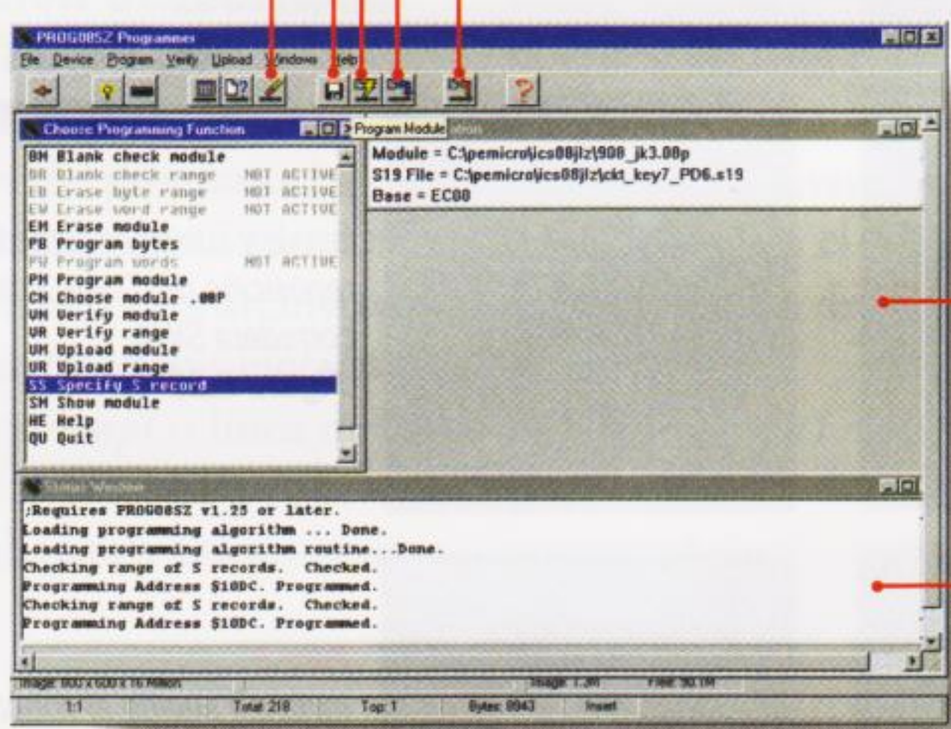
Selección del archivo S19

Haciendo clic en este icono se entra a programar el microcontrolador

Borrado del microcontrolador (Previa selección del algoritmo de borrado)

Verificación de que el microcontrolador ha quedado programado correctamente.

Lectura de la memoria del microcontrolador



Al hacer clic en el icono del rayo, el sistema graba el archivo S19 seleccionado en el paso anterior dentro de la memoria del microcontrolador. Una vez iniciado este proceso en la ventana de estado se visualiza el porcentaje progresivo en que se va cumpliendo el proceso de programación.

Ventana de estado

Figura 8.16 *Programación del microcontrolador*

El entorno **ICS08JLZ Development Kit**, proporcionado gratuitamente por la empresa Pemicro (www.pemicro.com), y sus programas de edición, ensamblaje, simulación y programación, constituyen herramientas muy importantes que pueden ser utilizadas por todos los interesados en iniciarse en la tecnología de los microcontroladores Motorola.

Aprovechando la *ROM Monitor* de estos microcontroladores, estos programas le permiten hacer no solo la tradicional simulación fuera de línea, sino también la simulación dentro del circuito. Con esta poderosa característica podemos depurar rápidamente nuestros programas, detectar errores y mirar el funcionamiento de la aplicación final.

A pesar de que dicha simulación dentro del circuito es un proceso que ejecuta lentamente cada una de las instrucciones escritas en un programa, si lo comparamos con un programa de emulación, sigue siendo muy importante para el usuario, quien puede visualizar el funcionamiento de sus algoritmos al ser aplicados dentro del circuito externo.

Ensamblaje

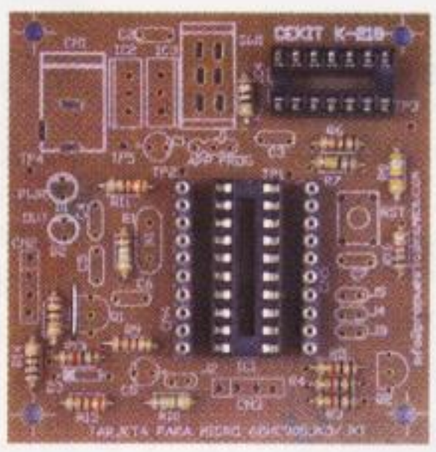
Figura 8.17 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el proyecto.*

Este proyecto constituye una importante herramienta para cualquier persona que desee inciarse en la tecnología Motorola. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

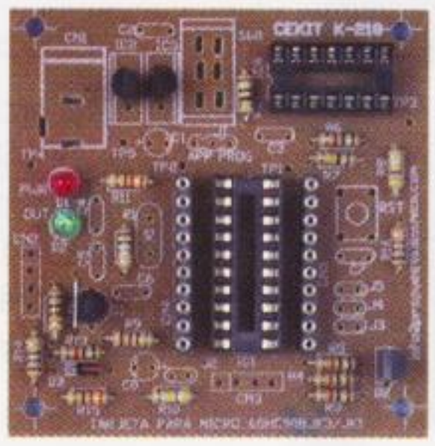
LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador MC68HC908JK3CP
1	IC2	Regulador 78L08
1	IC3	Regulador 78L05
1	IC4	Compuerta 74HC125
1	Q1	Transistor NPN 2N2222
1	Q2	Transistor PNP 2N2907
1	D1	Diodo LED rojo 3mm
1	D2	Diodo LED verde 3mm
1	D3	Diodo rectificador 1N4148
2	C1,C8	Condensadores electrolíticos 22uF/50V
4	C2,C3,C6,C7	Condensadores cerámicos 0.1uF/50V
2	C4,C5	Condensadores cerámicos 22pF/50V
1	X1	Cristal de cuarzo de 5Mhz
1	R1	Resistencia 10M 1/4 W
1	R2	Resistencia 10 Ohm 1/4W
6	R3,R4,R5,R6,R13,R16	Resistencias 10K 1/4W
1	R7	Resistencia 100K 1/4W
2	R8,R10	Resistencias 470K 1/4W
2	R11,R15	Resistencias 3.3K 1/4W
2	R12,R14	Resistencias 100 Ohm 1/4W
1	R9	Resistencia 2.2K 1/4W
1	SW1	Interruptor corredera de 2 polos 2 posiciones
1	RST	Pulsador pequeño para circuito impreso
1	CN1	Conector CA/CC
1	CN2	Conector en línea de 3 pines
1	CN3	Conector en línea de 4 pines
2	CN4, CN5	Conectores en línea tipo socket de 10 pines
1	J1	Conector para jumper de 3 pines en línea
4	J2,J3,J4,J5	Conectores para jumper de 2 pines en línea
1	K-218	Circuito impreso CEKIT
5	TP1,TP2,TP3,TP4,TP5	Espadines
1		Conector DB9 hembra normal con carcasa
1		Base para circuito integrado de 20 pines
1		Base para circuito integrado de 14 pines
5		Jumpers
1m		Cable ribbon de 3 hilos

Paso 1
Con un alambre delgado, instale y suelde los puentes marcados en el circuito impreso, instale y suelde también las resistencias, las bases y los conectores en línea CN4 y CN5.
Figura 8.18

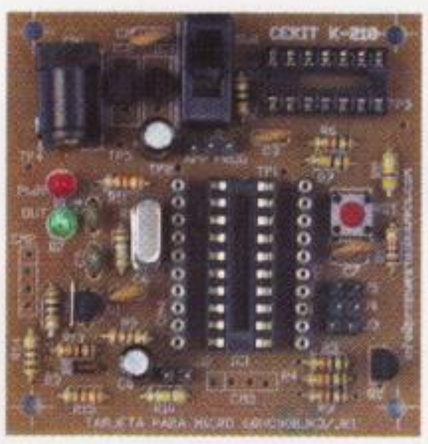


Paso 2
Luego instale y suelde los diodos LED, los transistores (Q1 y Q2) y los dos circuitos integrados reguladores de voltaje (IC2 e IC3) .
Figura 8.19



Paso 5
Instale y suelde el pulsador de reposición (RST) y el interruptor de corredera SW1.
Figura 8.22

Paso 3
Instale y suelde los condensadores cerámicos C2 a C7.
Figura 8.20



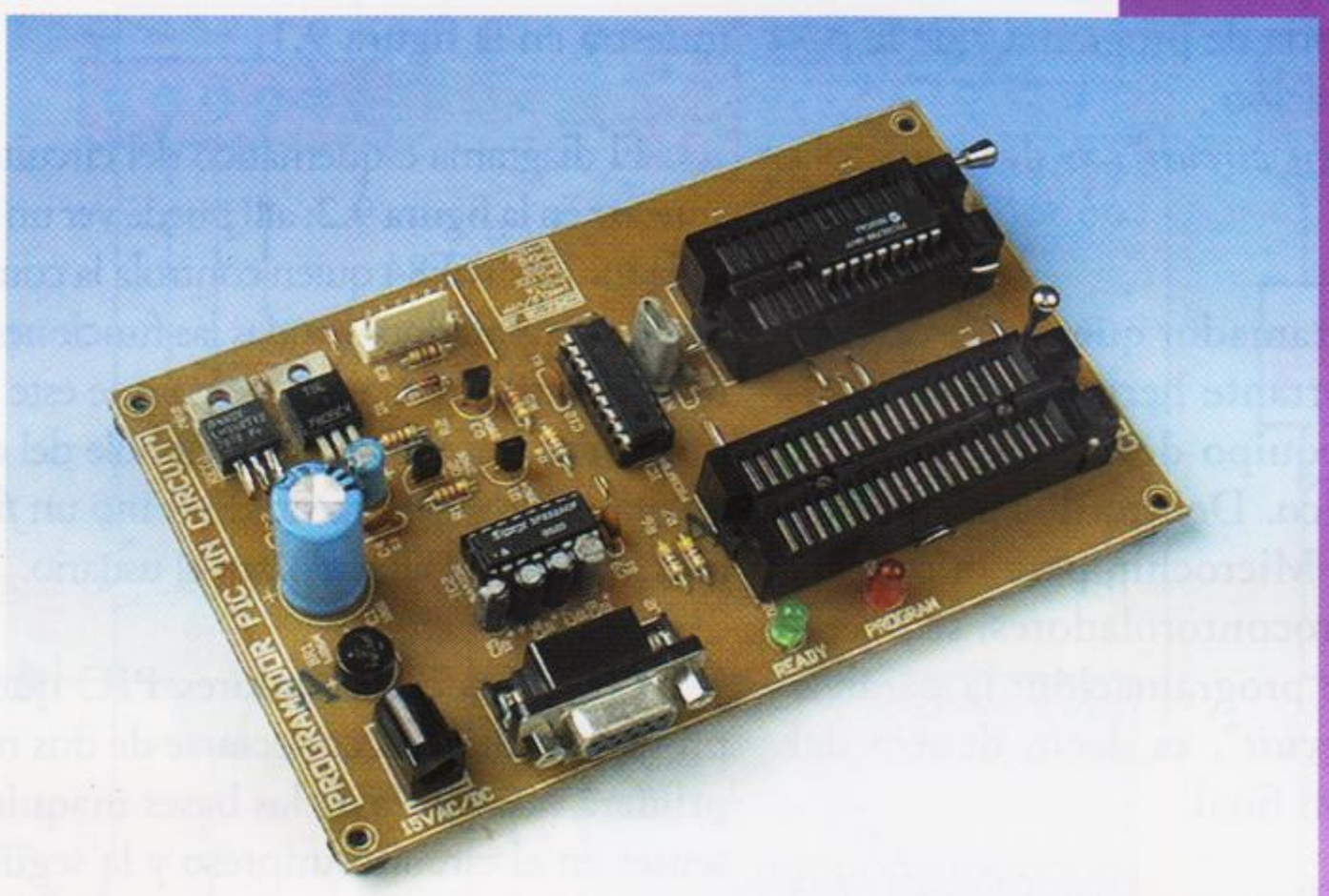
Paso 6
Instale y suelde el conector de alimentación CA/CC (CN1) y los conectores tipo cerca J1 a J5.
Figura 8.23

Paso 4
Instale y suelde los dos condensadores electrolíticos (C1 y C8) y el cristal de cuarzo de 5 MHz (X1).
Figura 8.21

Paso 7
Instale y suelde por último los dos conectores en línea CN2 y CN3.
Figura 8.24

Paso 8. Pruebas finales
Alimentar la tarjeta mediante un adaptador de 12 VCC y sin el microcontrolador JK3. La medida de voltaje entre TP4 y TP3 debe ser alrededor de los 5V y entre TP4 y TP2 alrededor de los 8V. Instale el microcontrolador e intente ejecutar el programa de simulación dentro del circuito configurando adecuadamente la ventana de entrada tal como se explicó anteriormente.

PROYECTO 9



PROGRAMADOR "In Circuit" DE MICROCONTROLADORES PIC

El desarrollo de sistemas electrónicos con microcontroladores requiere de tres pasos fundamentales: escribir el programa, ensamblarlo para crear un archivo binario, y un programador que tome ese archivo y lo almacene en la memoria del microcontrolador para que éste lleve a cabo las tareas dispuestas por el usuario. En este proyecto se construirá un programador de microcontroladores PIC y se analiza paso a paso su manejo. En general este procedimiento es muy similar para cualquier programador de cualquier marca.

Características

- Interfaz interactiva
- Acepta la variedad de microcontroladores PIC 16Cxx y 12Cxx
- Lectura y escritura de archivos en formato IN8HEX
- Chequeo de memoria vacía
- Lectura de memoria de programa y de la palabra de configuración
- Programación "*in circuit*", o dentro de la aplicación

Este es un programador que puede constituirse en una importante herramienta de trabajo en cualquier equipo de laboratorio o de desarrollo electrónico. Dentro de las especificaciones dadas por Microchip para la programación de sus microcontoroladores, se destacan dos técnicas de programación: la paralela y la serial o "*in circuit*", es decir, dentro del circuito o aplicación final.

Un programador "*in circuit*" se caracteriza por enviar el programa directamente a la memoria del PIC en cualquier sitio en donde se encuentre ubicado; de esta manera un aparato que se tenga previamente programado dentro de algún circuito, puede reprogramarse sin necesidad de removerlo del sitio donde hace su trabajo. Así el PIC puede estar soldado en forma permanente (sin base), simplemente conectando adecuadamente la tarjeta programadora al microcontrolador.

Esa es la principal diferencia con respecto a los programadores "STAND ALONE", independientes o paralelos, los cuales cuentan con una base tipo ZIF en la cual se inserta el PIC en el momento de programarlo, luego deben retirarlo del programador e insertarlo dentro del circuito.

Para utilizar este programador debe tener a disposición el prototipo montado, en el cual se conecta la tarjeta programadora, tal como se muestra en la **figura 9.1**.

El diagrama esquemático del circuito lo puede apreciar en la **figura 9.2**; allí puede ver un microcontrolador PIC16F84 quien controla la comunicación con la computadora y todas las funciones de los demás circuitos. Tenga en cuenta que este microcontrolador es parte fija e indispensable del circuito, su programa debe ser entendido como un *firmware*, el cual no debe ser alterado por el usuario.

Los microcontroladores PIC que se van a programar pueden conectarse de dos maneras: la primera es a través de las bases maquinadas presentes en el circuito impreso y la segunda es conectando la tarjeta al prototipo mediante un conector de seis pines, cada uno de los cuales debe ser conectado con el PIC objetivo, así:

1. **MCLR.** Debe conectarse al pin de *reset* del microcontrolador que se va a programar.
2. **DATA.** Por este pin van los datos de programación y debe conectarse al pin RB7 del PIC objetivo.
3. **CLOCK.** Es la señal de reloj, debe conectarse al pin RB6 del PIC que se va a programar.
4. **GND.** Es la tierra del programador, debe conectarse a la tierra del prototipo.
5. **GND.** Tierra.
6. **VCC.** Salida de 12V.

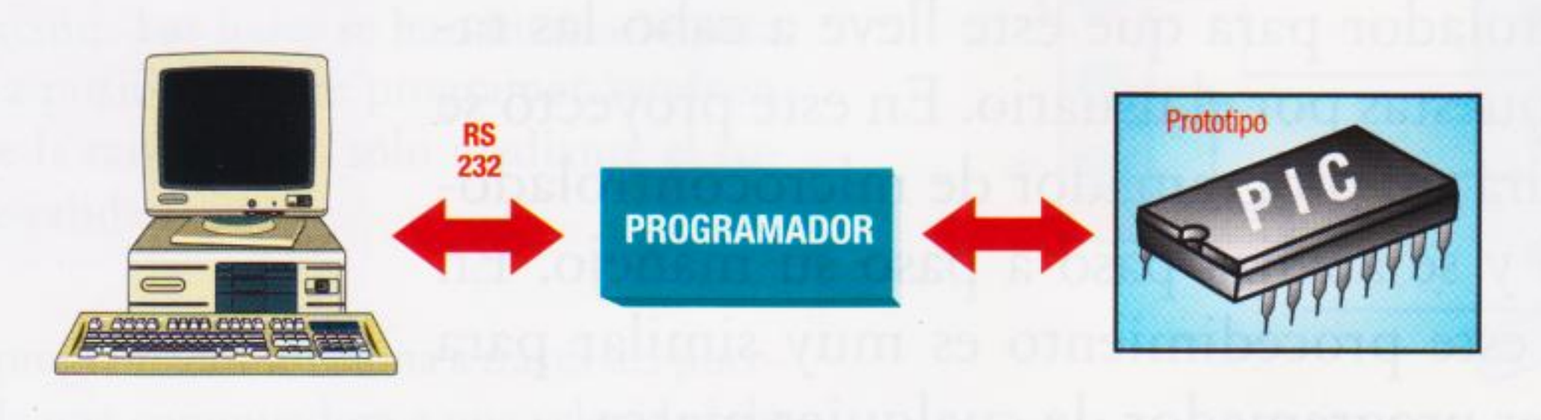


Figura 9.1 *Esquema de conexión del programador*

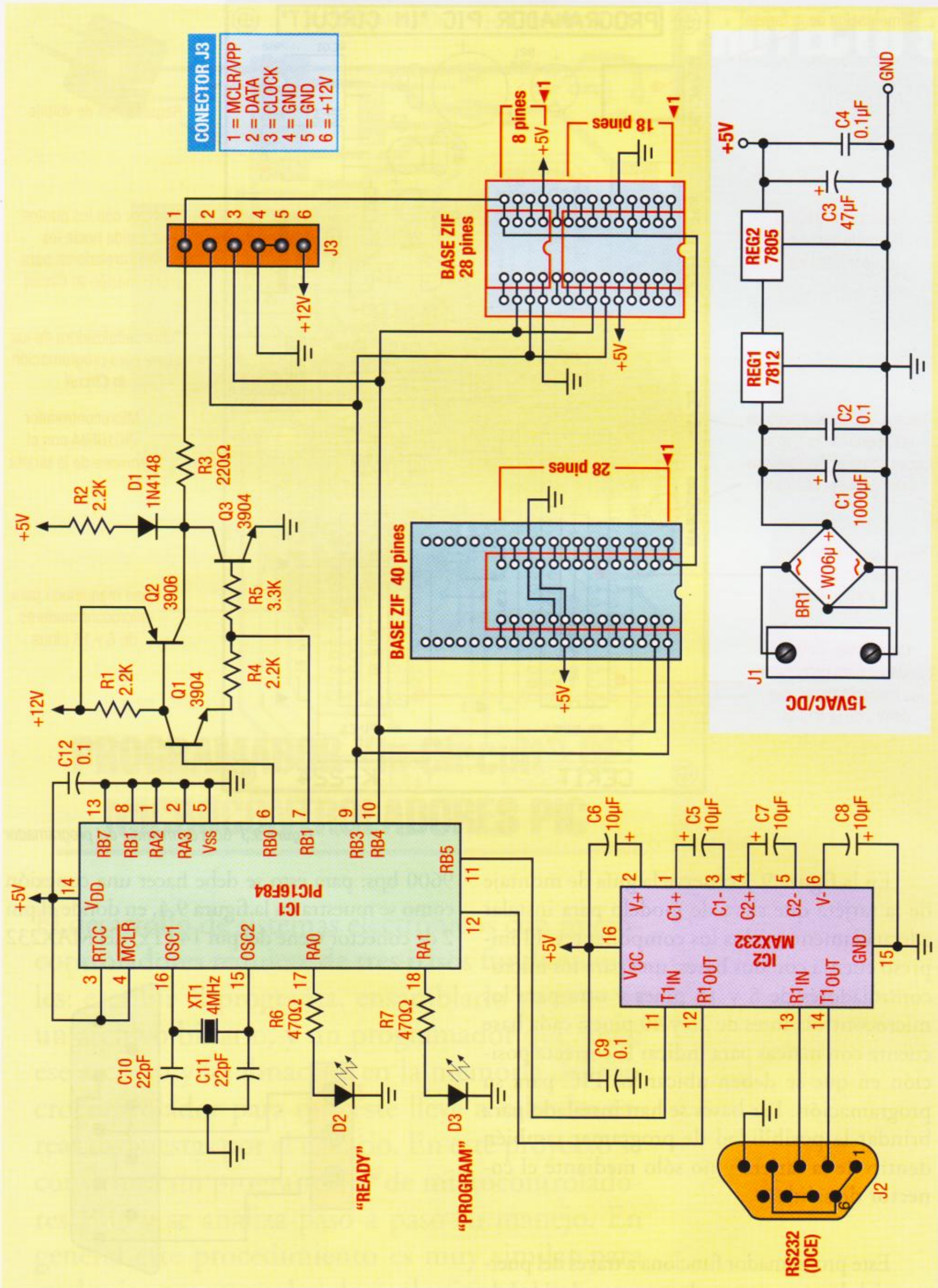


Figura 9.2 *Diagrama esquemático del programador "in circuit"*

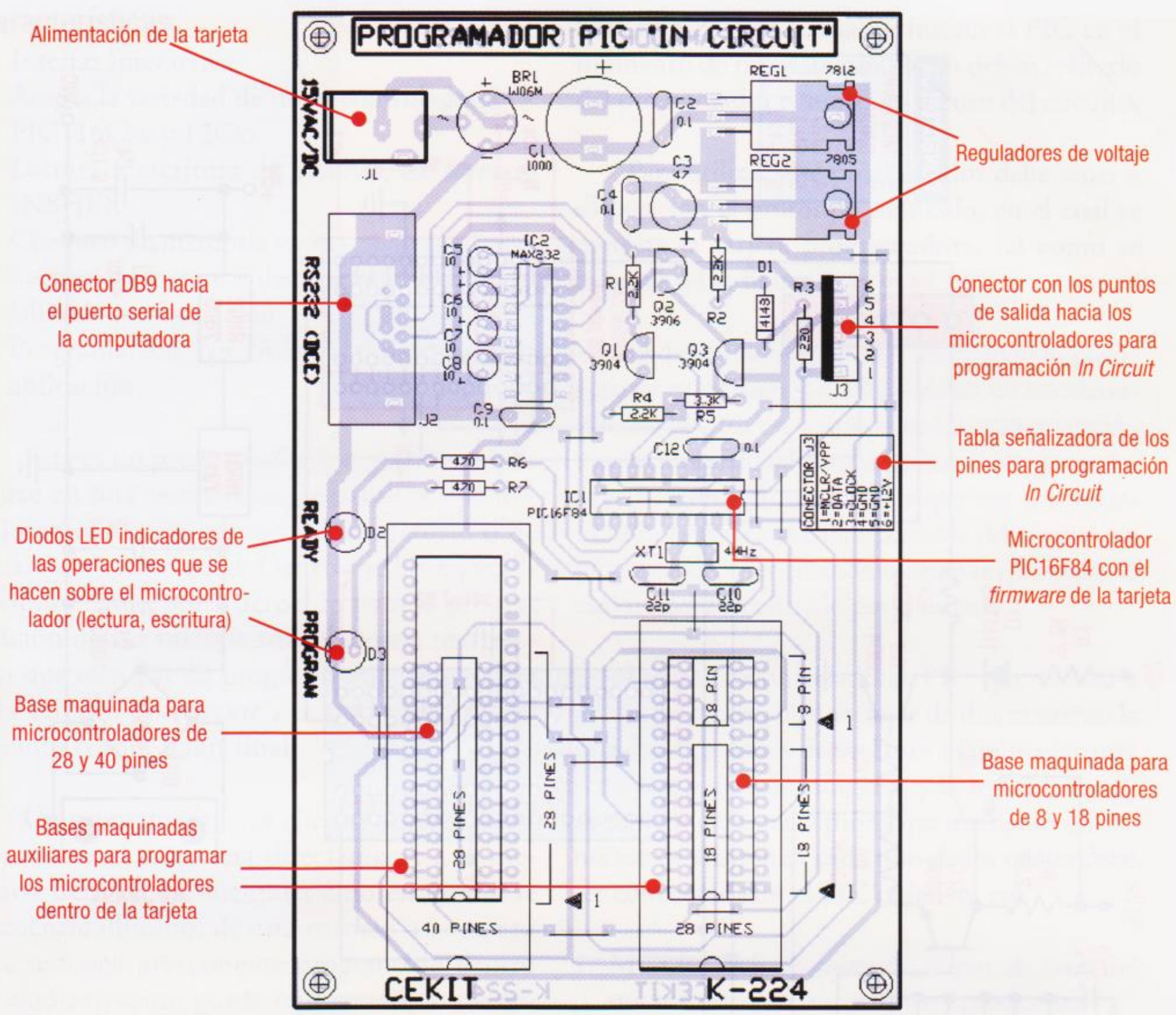


Figura 9.3 *Guía de montaje del programador*

En la **figura 9.3** observe la guía de montaje de la tarjeta que sirve de modelo para instalar adecuadamente todos los componentes. El impreso cuenta con dos bases: una para los microcontroladores de 8 y 18 pines y otra para los microcontroladores de 28 y 40 pines; cada base cuenta con marcas para indicar la correcta posición en que se deben ubicar los PIC para su programación. Las bases se han instalado para brindar la posibilidad de programar también dentro de la tarjeta y no sólo mediante el conector de salida.

Este programador funciona a través del puerto serial de una computadora a una velocidad de 9600 bps; para esto se debe hacer una conexión como se muestra en la **figura 9.4**, en donde el pin 2 de conector viene del pin 14 (Tx) del MAX232

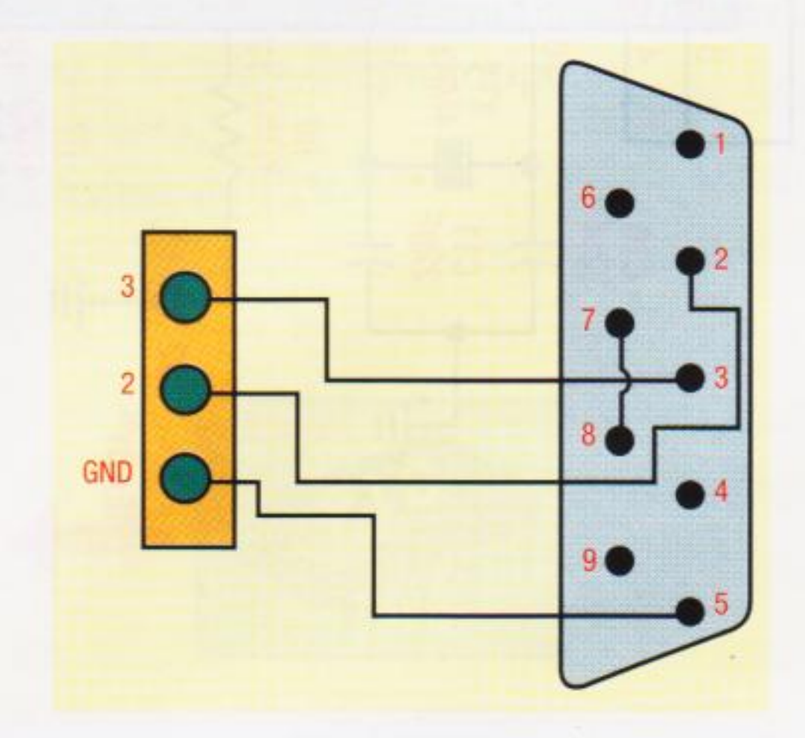


Figura 9.4 *Conexión con la computadora*

```
> progra  /COM1
> ERROR, PROGRAMADOR NO RESPONDE!
```

Figura 9.5 *Al igual que en cualquier programador comercial convencional, si existe algún problema en la detección de la tarjeta, el programa no funcionará*

y el pin 3 del conector viene del pin13 (Rx) del MAX232. Sin embargo, para que usted se despreocupe de esta conexión, ella ya viene hecha en el DimB9 instalado en la tarjeta; de esta manera el cable de conexión entre la tarjeta y la computadora es un cable DB9- DB9 normal conectado uno a uno.

Instalación del software y guía rápida de programación

El *software* del programador está escrito en lenguaje C para DOS y el ejecutable es un archivo por lotes (*batch*); debe ejecutarlo desde un *prompt* o línea de comandos DOS. El programa puede trabajar en máquinas que tengan los sistemas operativos **DOS, Windows 3.1/95/98** ó **OS/2**; y en máquinas que posean procesadores Intel del 386 en adelante.

Para instalar el programa sólo basta copiarlo en alguna carpeta del disco duro de la computadora; no altera ningún registro del sistema operativo y de la misma manera, para retirarlo sólo basta borrar el archivo.

Para ejecutar el programa debe digitar en el símbolo indicador (*prompt*) del DOS:

> *progra /com1* (o com2, según el puerto donde lo tenga conectado).

Si la tarjeta no se encuentra conectada al puerto serial o el *firmware* ha sido alterado, no se establecerá comunicación y el programa no se ejecutará, **figura 9.5**.

Si la conexión entre la PC y la tarjeta, al igual que el cable serial están bien instalados, el *firmware* del microcontrolador podrá llevar a cabo un proceso de reconocimiento (*hand-shaking*) y podrá entrar a la pantalla inicial del programa, **figura 9.6**.

En esta pantalla se indican: el PIC actualmente seleccionado, el contenido del *buffer* de memoria (si éste aún no se ha cargado aparecerá el aviso de "no definido"), la palabra de configuración y un menú de opciones. Para seleccionar cada una de las posibilidades de este menú, basta con teclear la letra que se encuentra a la izquierda de cada opción entre paréntesis; analicemos cada uno de los ítem de este menú.

Figura 9.6 *Si no existe ningún problema en las conexiones, el programa inicia su operación*

Referencias de los distintos microcontroladores aceptados por la tarjeta. A través del teclado se van mostrando secuencialmente las distintas posibilidades.

En esta línea tiene el tipo de PIC seleccionado en el momento. Haciendo clic en la tecla *enter* lo seleccionamos definitivamente para trabajarlo con la tarjeta.

Justo en frente de cada microcontrolador puede visualizar el tamaño de su memoria de programa.

Figura 9.7 *La selección del tipo de PIC que se va a programar se hace a través de las flechas del teclado, moviéndose hacia arriba o hacia abajo hasta llegar al PIC seleccionado; en ese momento, se debe presionar la tecla ENTER*

Seleccionar el tipo de PIC

Este programador puede trabajar con toda la familia de microcontroladores PIC 16XXX. Con esta se opción selecciona el PIC específico con el cual se va a trabajar, **figura** 9.7. Al presionar la tecla T, sale una lista de todos los PIC disponibles. Para esa selección, debe buscar el dispositivo correcto con las teclas de flecha hacia arriba y flecha hacia abajo o las teclas **Repág.** y **Avpág**. En esta pantalla aparece a la izquierda el tipo de PIC y a la derecha el tamaño de la memoria de programa de cada uno de ellos. Los tipos de microcontroladores PIC que puede trabajar con esta tarjeta se muestran en la **tabla 9.1**; sin embargo, en versiones posteriores de este producto se abarcarán otros modelos.

Leer PIC

Si se teclea la letra R, la tarjeta leerá el contenido de la memoria del microcontrolador objetivo que se encuentre conectado a la tarjeta. Este proceso se demora unos cuantos segundos, al cabo de los cuales aparece un nuevo menú, donde se ven nuevas opciones, **figura 9.8**. La lectura del programa del circuito integrado ha quedado almacenada en un espacio de memoria llamado imagen de entrada. Entre las nuevas opciones que aparecen se encuentran la de abrir un archivo de disco, la de escribir en el PIC y establecer la palabra de configuración. Cuando llevemos a cabo este procedimiento, en la pantalla se mostrará una barra indicadora (con base en puntos) que irá aumentando proporcionalmente con la lectura de la memoria del microcontrolador.

Tecleando la letra D puede visualizar en la pantalla todo lo que se leyó desde la memoria del microcontrolador

Tecleando la letra R puede leer la memoria del microcontrolador. Este proceso es un poco lento en comparación con los programadores paralelos. Una vez iniciado este proceso puede ver en la parte inferior de la pantalla una barra indicadora.

Barra indicadora del porcentaje leído del PIC. Ella va aumentando a medida que lee la memoria del microcontrolador.

Al terminar la lectura de la memoria del microcontrolador se despliega un mensaje indicando la cantidad de palabras leidas.

Figura 9.8 *El sistema puede leer el contenido de la memoria de los PIC. Este proceso se demora un poco más, comparado con los programadores paralelos.*

Tipos de PIC soportados por la tarjeta								
12C508	12C509	16C554	16C556	16C558	16C61	16C62	16C62A	16C62B
16C620	16C620A	16C621	16C621A	16C622	16C622A	16C63	16C63A	16C64
16C64A	16C65	16C65A	16C65B	16C66	16C67	16C71	16C710	16C711
16C72	16C72A	16C73	16C73A	16C73B	16C74	16C74A	16C74B	16C76
16C77	16C84	16F83	16F84	16C923	16C924	16CE623	16CE624	

Tabla 9.1 *Tipos de PIC soportados por la tarjeta*

El sistema permite que los datos leídos de la memoria del microcontrolador sean almacenados en disco, para que puedan ser utilizados en otra oportunidad con el fin de programar otros microcontroladores. Para guardar el archivo hay que oprimir la tecla S, allí el programa desplegará una línea de comandos donde debe especificar el nombre del archivo con el que desea guardar los datos en el disco, **figura 9.9**.

Mostrar *Buffer*

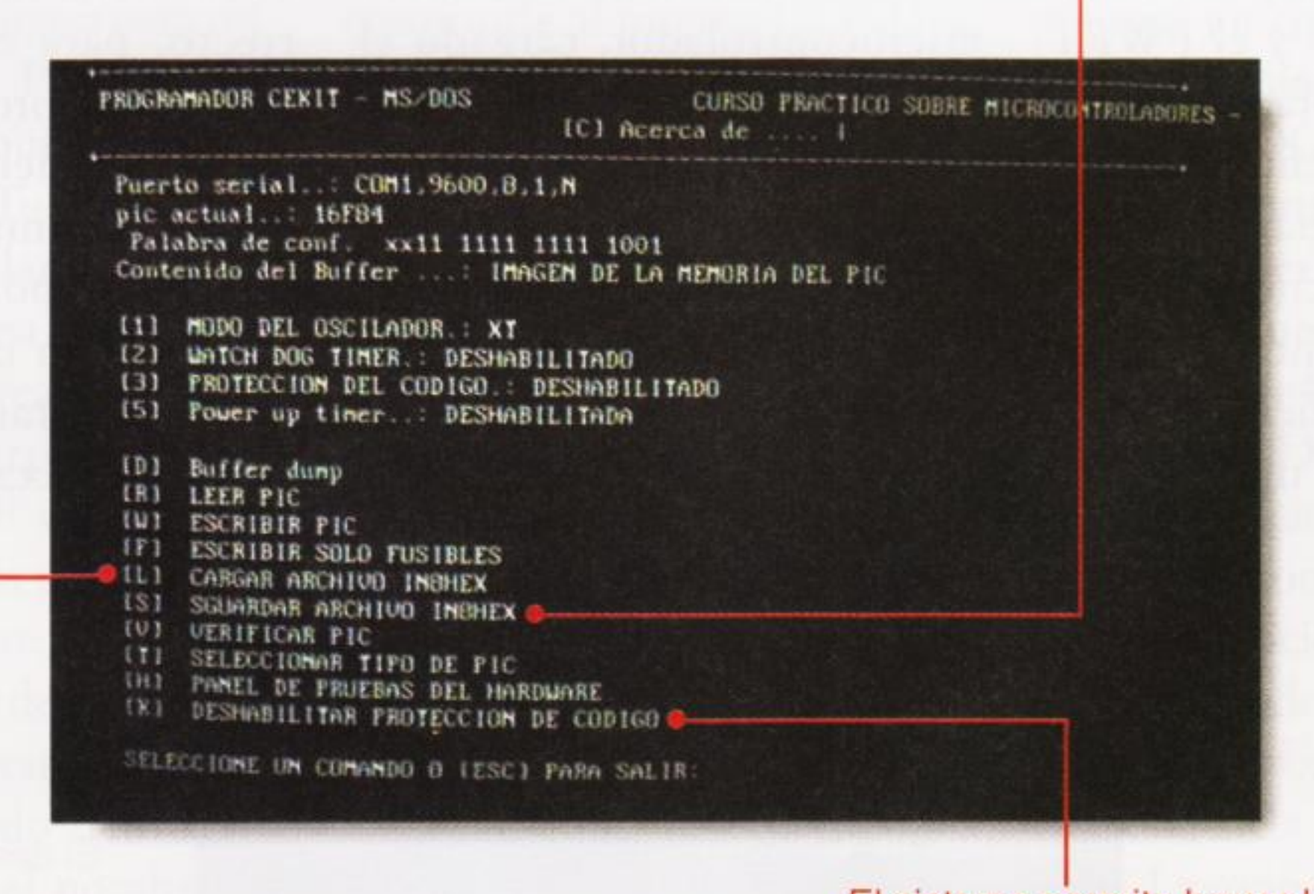
Esta opción se habilita al teclear la letra **D**. El *buffer* o memoria auxiliar se utiliza para almacenar temporalmente un programa en hexadecimal, el cual puede provenir de la lectura del PIC o de un archivo grabado en disco. Este *buffer* se puede visualizar con esta opción, con la cual se muestran todas las direcciones de memoria y su contenido, **figura 9.10**.

Programación del microcontrolador

El primer paso para programar un microcontrolador es cargar el archivo hexadecimal resultante de ensamblar el código fuente. Este archivo hexadecimal debe estar en una de las unidades de la computadora; si pulsa la tecla L, el programa le permitirá cargar el archivo hexadecimal que desea programar, para lo cual pide la ubicación y el nombre del archivo, **figura 9.11**. Cuando el archivo ha sido leído, queda almacenado en el *buffer* del programa, por lo tanto puede ser visualizado por medio de la letra D.

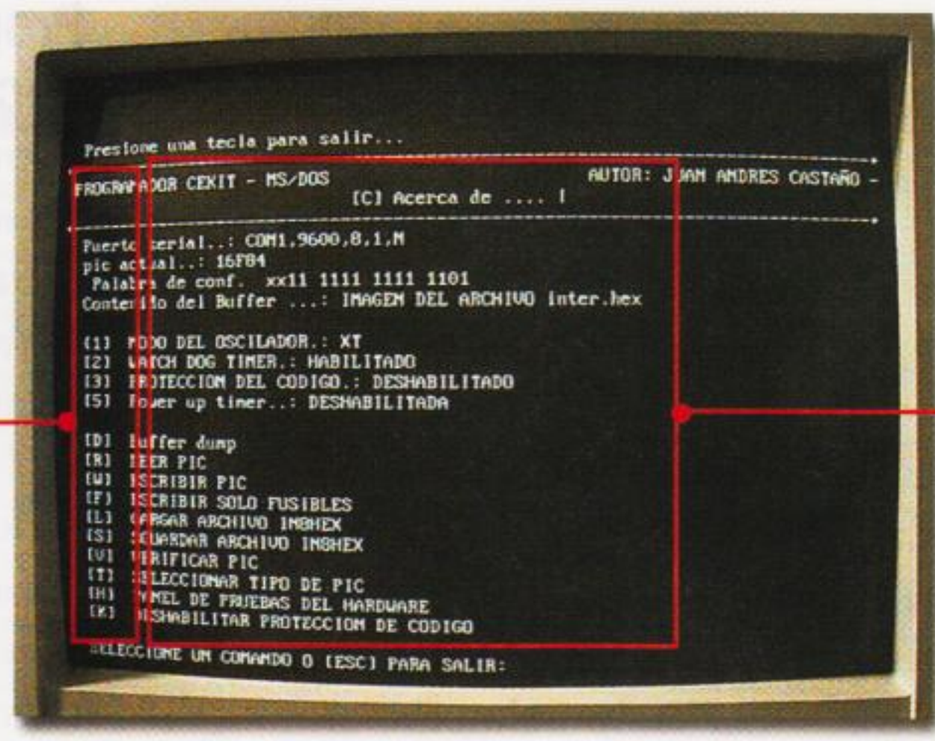
Pulsando la tecla **L** puede cargar un archivo almacenado en el disco duro, para programar cualquier microcontrolador

Este sistema permite que una vez leído el contenido de la memoria de un microcontrolador PIC, usted pueda grabar esa información dentro de un archivo del disco duro. Para esto se debe pulsar la letra **S**



El sistema permite borrar los contenidos de la memoria de un microcontrolador PIC. Para ejecutar esa tarea se debe pulsar la letra **K**.

Figura 9.9 *Guardando en disco el archivo binario leído desde el microcontrolador*



Posiciones de memoria

Contenido de cada posición

Figura 9.10 *Visualización del archivo leído*

Después de que se ha cargado el archivo hexadecimal, se deben establecer los denominados **fusibles de programación**; ellos corresponden a las condiciones bajo las cuales va a funcionar el sistema, **figura 9.12**. Esos parámetros de configuración son: el tipo de oscilador que se va a instalar, el estado del temporizador *watchdog*, la habilitación del código protegido y el PWRT (ver lección 3 de la sección de teoría). En el menú aparecen estas opciones para que las modifique de acuerdo a sus necesidades; como se puede observar, a la izquierda de cada fusible aparece un número ([1], [2], [3] y [4]), oprimiendo cada número se modificará cada uno de los fusibles. Por ejemplo, si el oscilador se encuentra en RC y oprimime la tecla 1 este fusible cambiará a XT, si lo oprime de nuevo cambiará a INTRC y así sucesivamente para cada una de las opciones.

Una vez ha seleccionado el microcontrolador, cargado el archivo hexadecimal y configurado adecuadamente los fusibles, puede proceder a programar. Para llevar a cabo este proceso hay que oprimir la tecla W, con la cual se inicia el almacenamiento del programa en la memoria EEPROM. En este momento puede apreciar en la pantalla como van apareciendo poco a poco las distintas posiciones de memoria llenándose con los distintos datos del archivo hexadecimal, **figuras 9.13a** y **9.13b**. El programa hace un procedimiento de verificación para comprobar que en cada posición se ha almacenado el dato correcto, para ello compara el dato actual programado con el dato leído del *buffer*. Al final de la programación brinda un pequeño reporte donde especifica si han ocurrido errores o si la programación se desarrolló correctamente.

Cuando selecciona la opción de cargar un archivo desde el disco, el programa pone a su disposición una línea de comandos en donde debe teclear el nombre y la ubicación del archivo deseado

Si desea cancelar la operación de cargar un archivo, puede pulsar la tecla **ESC**

Figura 9.11 *Cargando el archivo hexadecimal desde el disco*

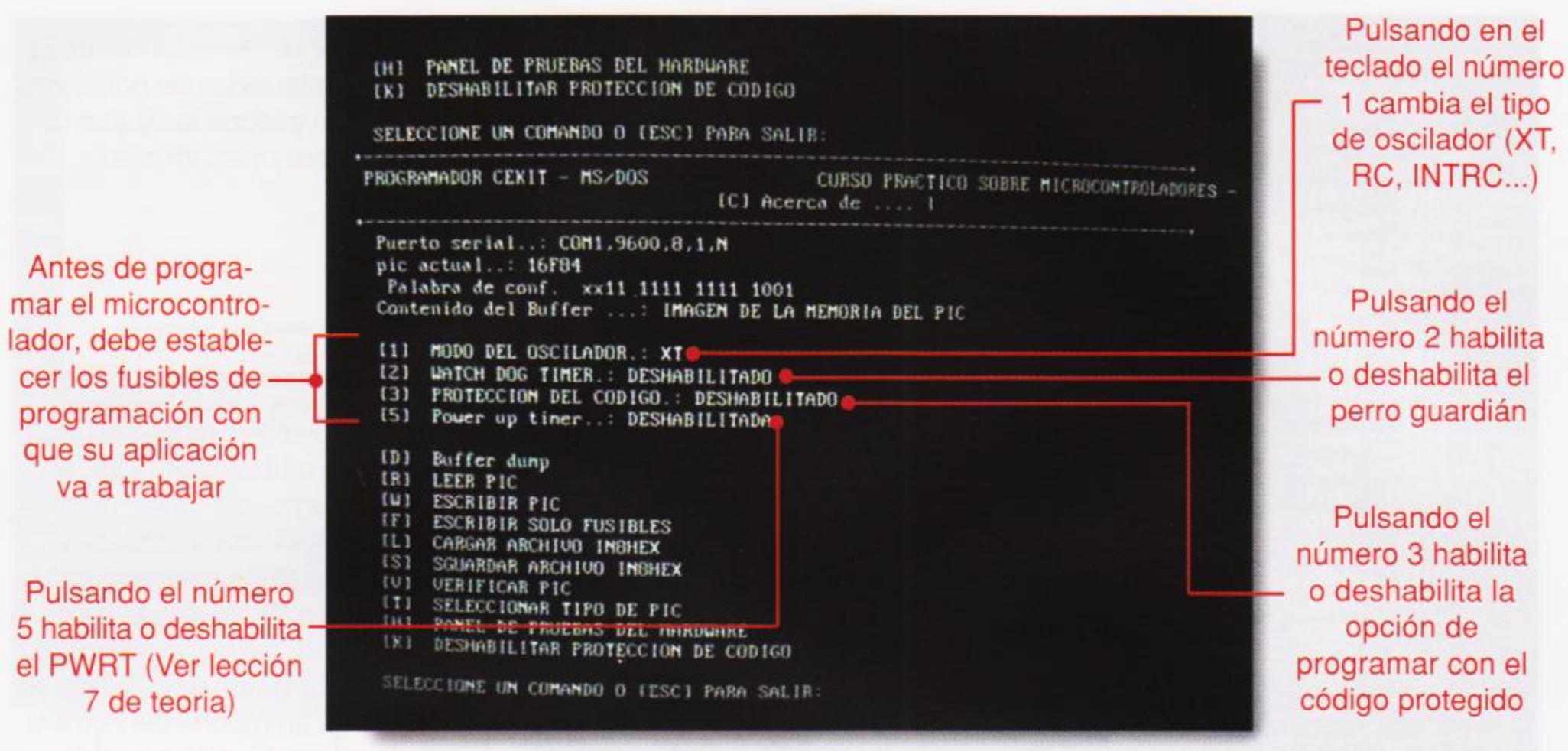


Figura 9.12 *Configuración de los fusibles de programación*

Si ha ocurrido un error en la programación de algunas de las posiciones de memoria, aparecerá un asterisco (*) enseguida de la posición donde se ha presentado la falla.

Finalmente es posible realizar una verificación total del programa grabado en el microcontrolador, comparándo este último con el *buffer* de memoria mantenido en la PC. Esta es una prueba que siempre se encontrará en cualquier programador y que le servirá para comprobar que el PIC ha quedado programado correctamente, **figura 9.14.**

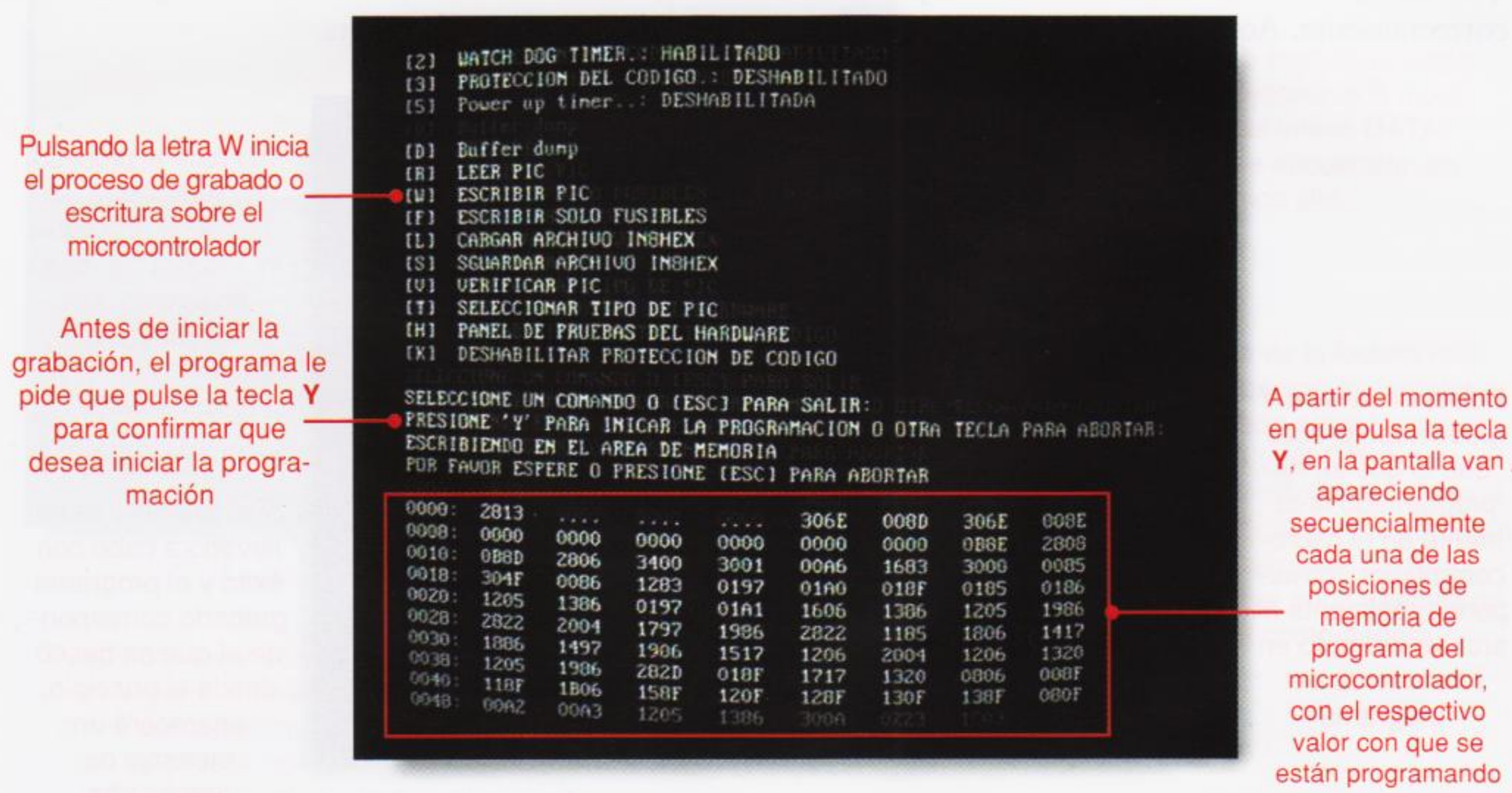


Figura 9.13a *Visualización del llenado de la memoria EEPROM*

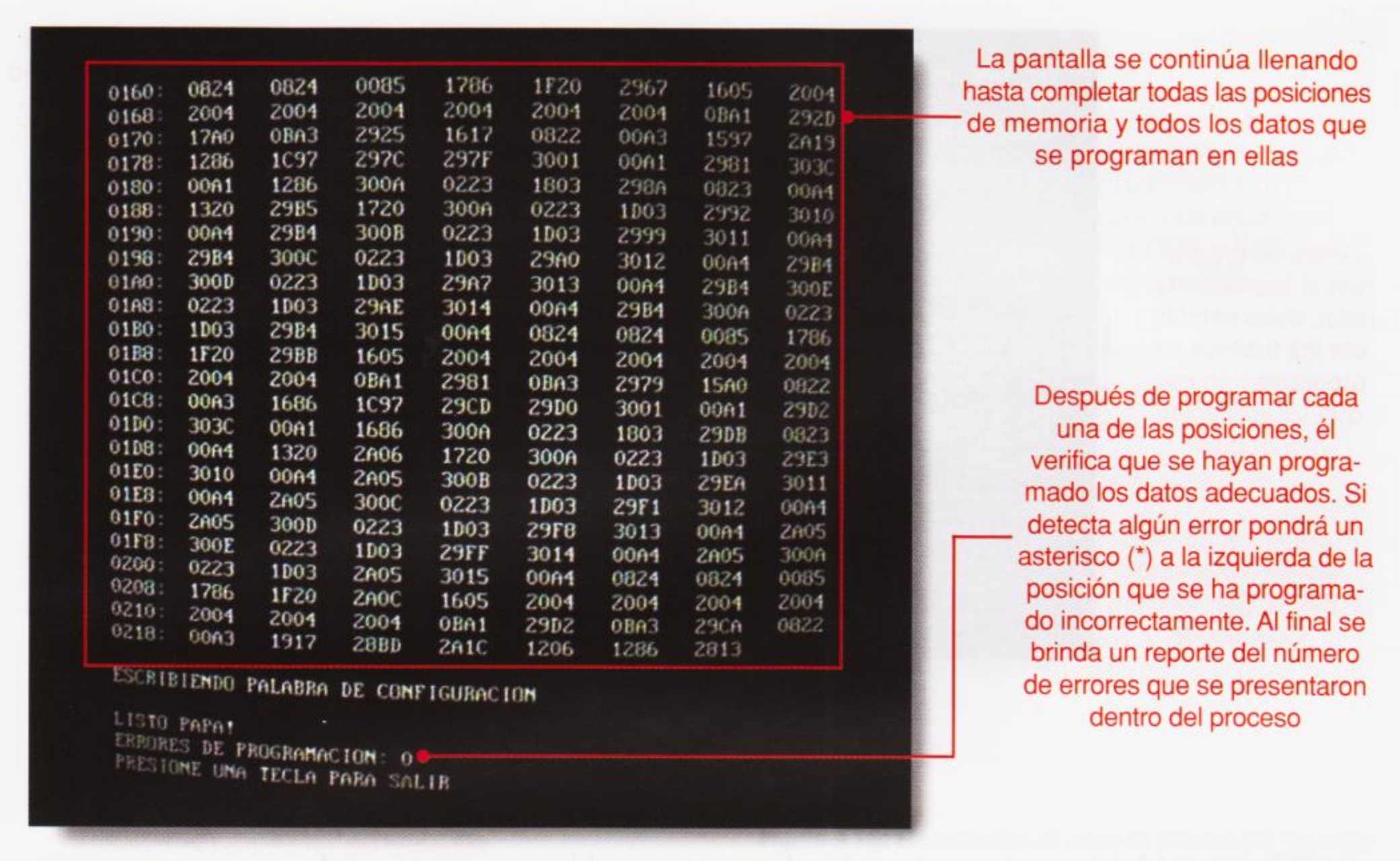


Figura 9.13b *Visualización del llenado de la memoria EEPROM*

Panel de pruebas del hardware

Con esta opción se practican una serie de pruebas para comprobar que la tarjeta está funcionando correctamente. Aquí entra a un nuevo menú en el cual le permite variar los voltajes del conector que va hasta el PIC objetivo, **figuras 9.15a, 9.15b** y **9.15c**. En esta presentación se muestran cinco tipos de pruebas distintas las cuales modifican los voltajes en cada uno de los pines del conector; leyendo con un multímetro los voltajes en cada uno de ellos, puede detectar alguna falla en la tarjeta.

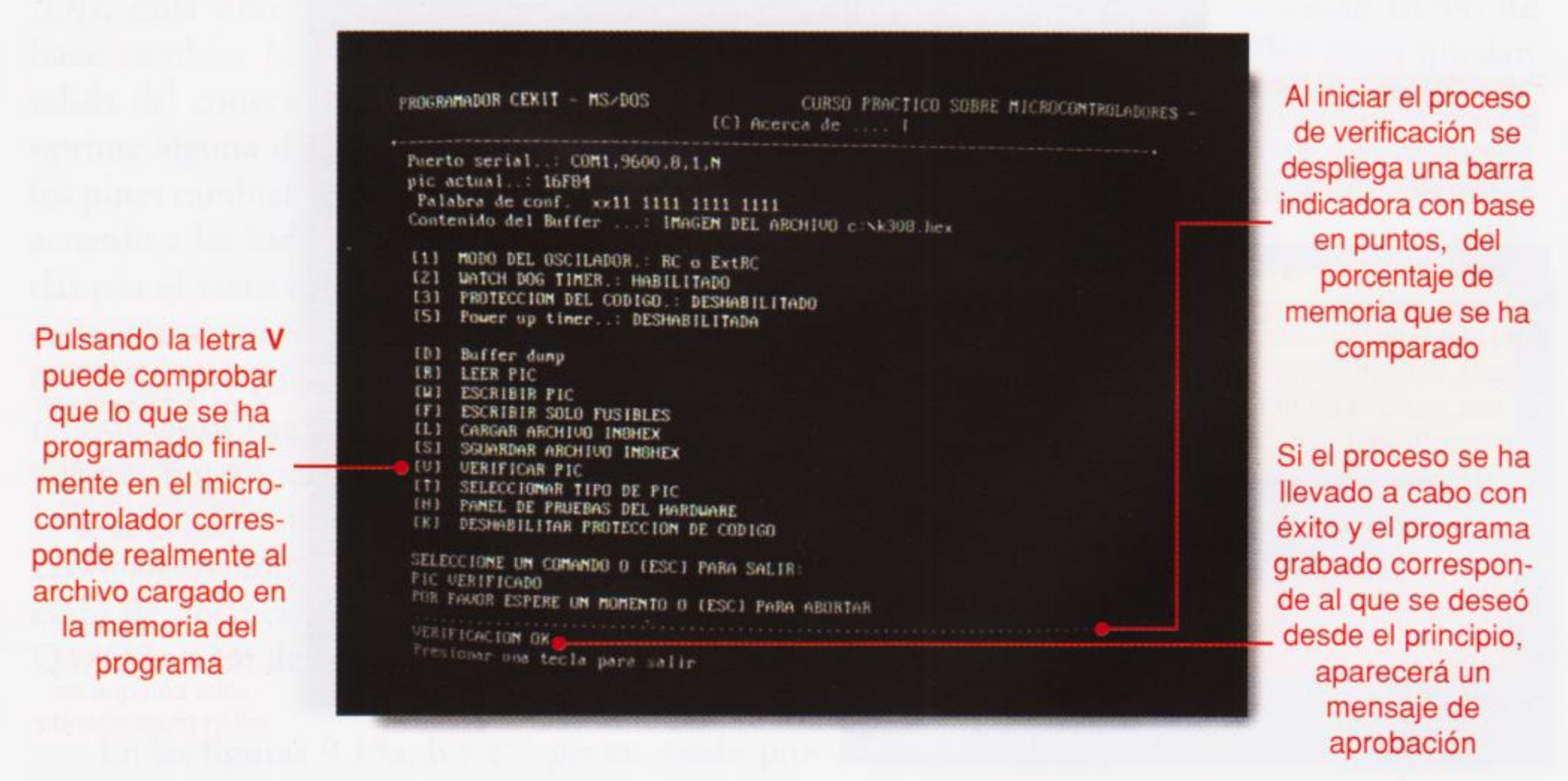
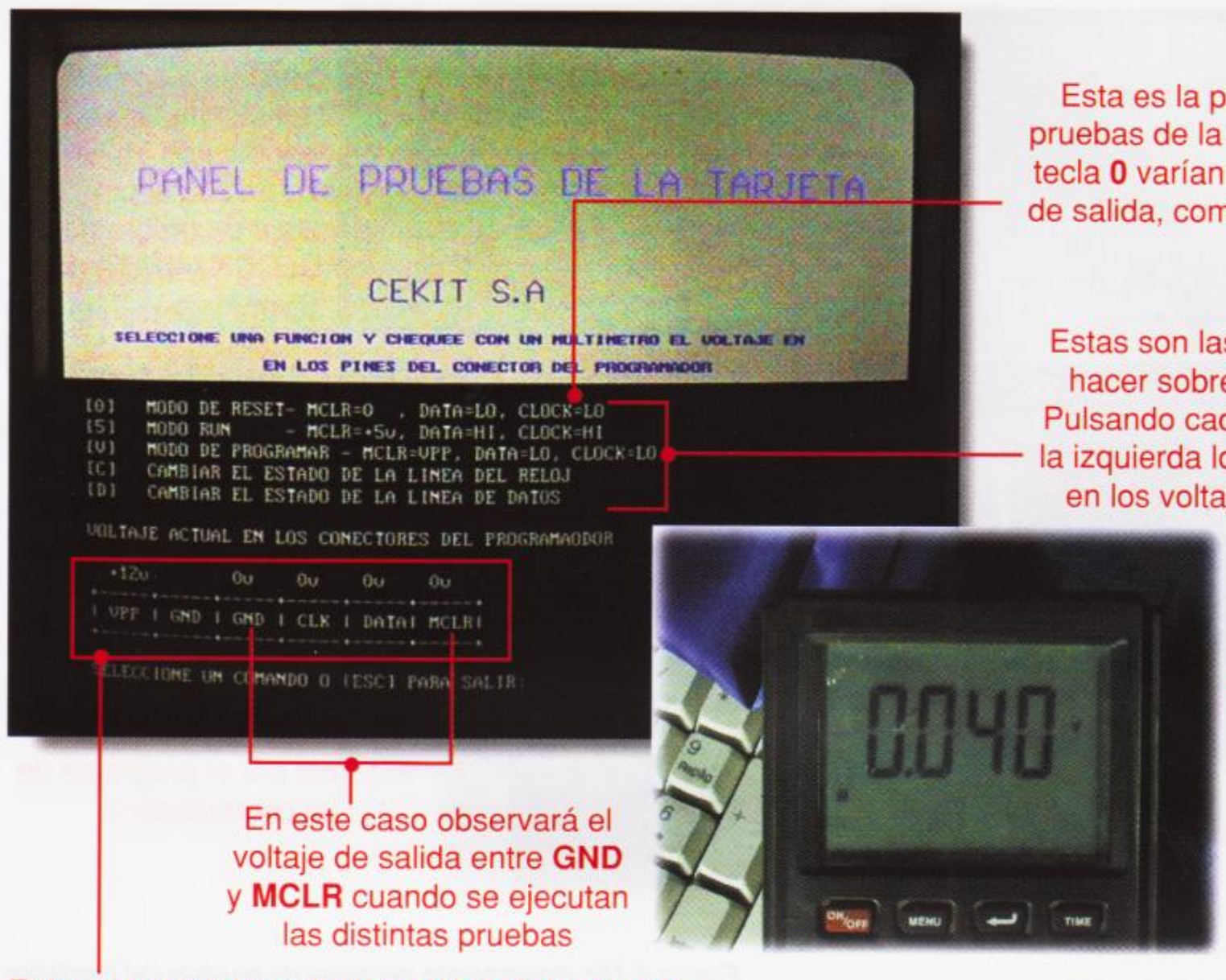


Figura 9.14 *Verificación de un correcto proceso de programación*



Esta es la presentación del panel de pruebas de la tarjeta. Haciendo clic en la tecla **0** varían los voltajes en el conector de salida, como lo indican los voltajes en esta línea

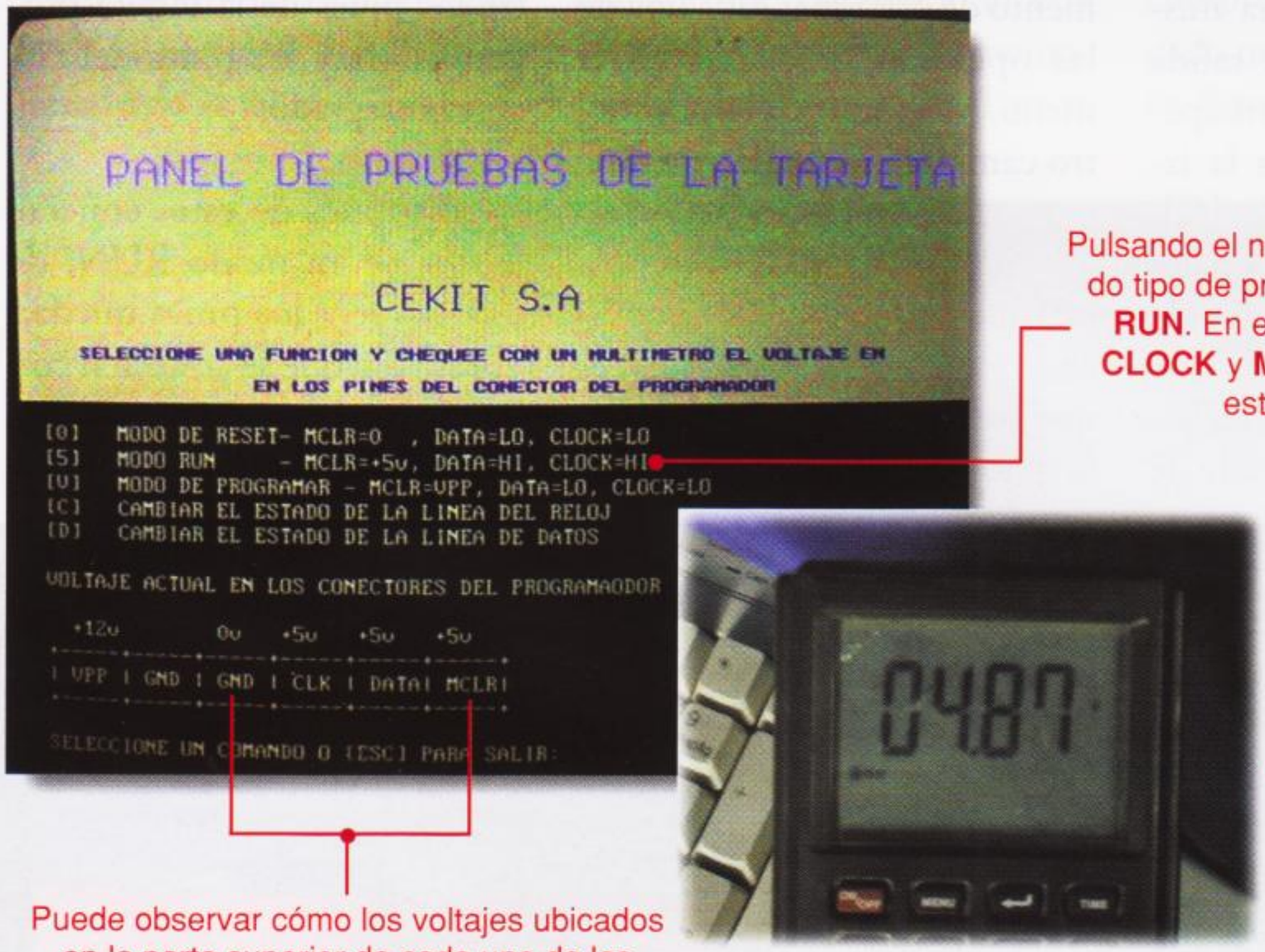
Estas son las cinco pruebas que puede hacer sobre el conector de la tarjeta. Pulsando cada uno de los caracteres de la izquierda logra cinco pruebas distintas en los voltajes de salida de la tarjeta.

Este es el voltaje de salida presente entre los pines **GND** y **MCLR** cuando oprime la tecla **0**

En este caso observará el voltaje de salida entre **GND** y **MCLR** cuando se ejecutan las distintas pruebas

Esta es una representación gráfica del conector de la tarjeta donde se muestran cada uno de los pines y su correspondiente voltaje de salida

Figura 9.15a *Presentación del panel de pruebas del hardware al pulsar la tecla 0*

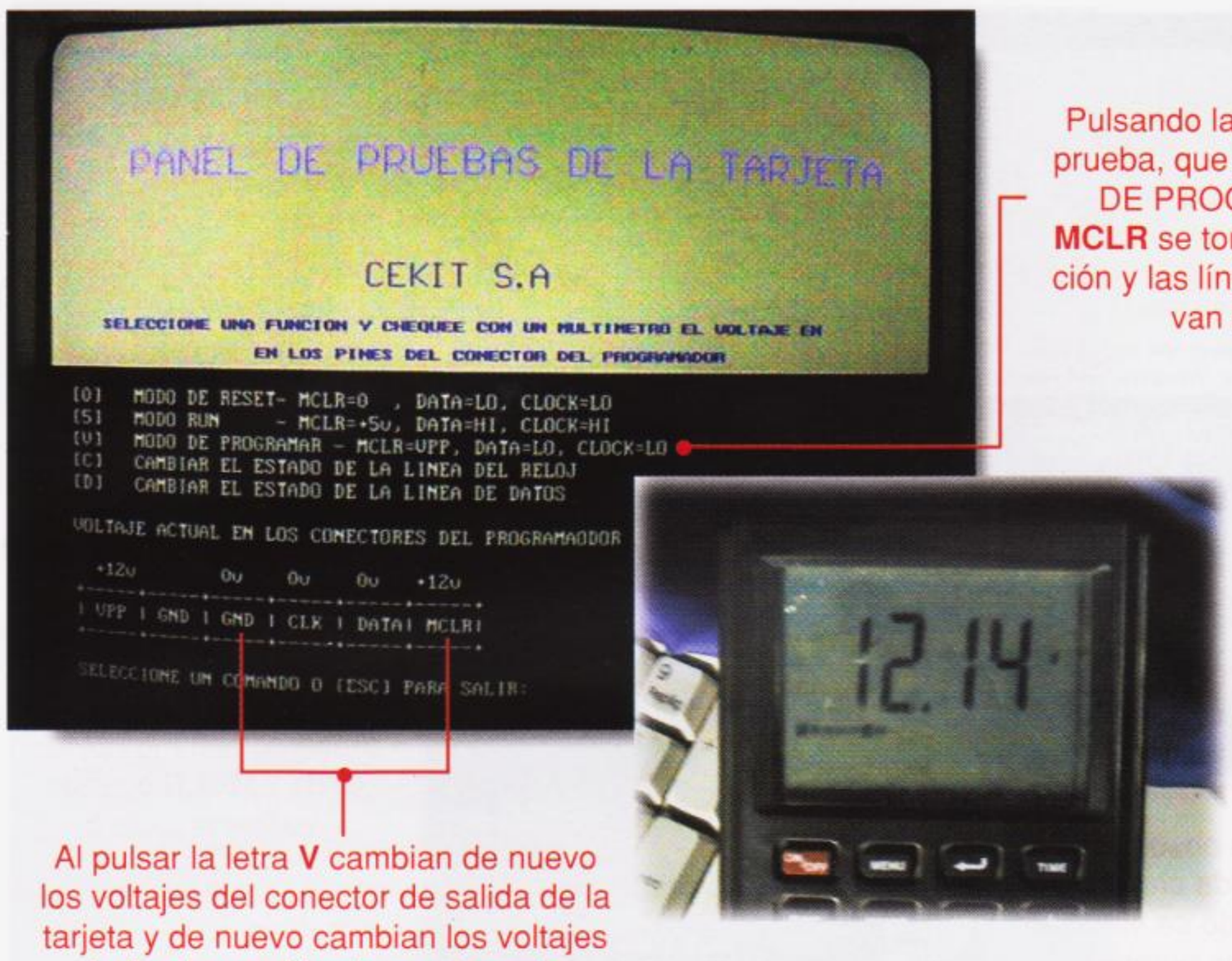


Pulsando el número 5 ejecuta el segundo tipo de prueba denominado modo **RUN**. En el cual las líneas **DATA**, **CLOCK** y **MCLR** se encuentran en estado lógico alto.

Esta es la lectura real medida con multímetro y tomada entre los pines **GND** y **MLCR**. Al tomar esta medida verifique que coincide con lo indicado en la gráfica del conector, presente en la pantalla de la computadora.

Puede observar cómo los voltajes ubicados en la parte superior de cada uno de los pines del conector cambian de acuerdo al tipo de prueba que ejecutó. Ésta es una indicación visual que facilita al usuario corroborar que los voltajes de su tarjeta se comportan correctamente

Figura 9.15b *Presentación del panel de pruebas del hardware al pulsar la tecla 5*



Pulsando la tecla **V** ejecuta la tercera prueba, que se ha denominado MODO DE PROGRAMAR. En ella el pin **MCLR** se toma el voltaje de programación y las líneas de **CLOCK** y **DATA** se van a nivel lógico bajo.

Este es el voltaje de programación de un microcontrolador. Esta lectura, con el multímetro, es tomada de nuevo entre los pines **GND** y **MCLR**, y coincide con lo indicado por el programa de la computadora.

Al pulsar la letra **V** cambian de nuevo los voltajes del conector de salida de la tarjeta y de nuevo cambian los voltajes indicadores ubicados en la parte superior del diagrama del conector.

Figura 9.15c *Presentación del panel de pruebas del hardware* ***al pulsar la tecla V***

En esta pantalla se puede apreciar un menú y una ilustración del conector de salida de la tarjeta. En el menú aparecen unos caracteres a la izquierda ([0], [5], [V], [C], [D]), cada uno de los cuales hace cambiar los voltajes de salida del conector. Cada que oprime alguna de estas teclas, los pines cambian de estado de acuerdo a las indicaciones dadas por el texto que le sigue a cada uno de esos caracteres y en el diagrama del conector se reflejan estos cambios. Si los voltajes no coinciden con los leídos en el conector, puede existir algún problema en el circuito de potencia (transistores Q1, Q2 y Q3) de la tarjeta.

En las **figuras 9.15a, b** y **c** se ha tomado la lectura entre los pines de GND y MCLR, al momento de presionar cada una de las opciones presentes en el menú, la lectura en el multímetro cambiará inmediatamente.

Sin lugar a dudas, usted pensará que se van a desperdiciar muchos pines del microcontrolador que pueden ser vitales dentro de la aplicación. Para evitar este contratiempo la solución es muy sencilla: basta colocar en el prototipo final una serie de *jumpers* con los cuales se selecciona el modo de trabajo de la tarjeta; es decir, cuando va a programarlo hay que colocar el *jumper* en modo programación, haciendo las conexiones necesarias entre los pines de la tarjeta programadora y los pines del circuito integrado.

Después de esto, conecte el *jumper* en modo RUN; de esta manera los pines quedan libres para que se conecten con los dispositivos externos deseados. Algunos microcontro-

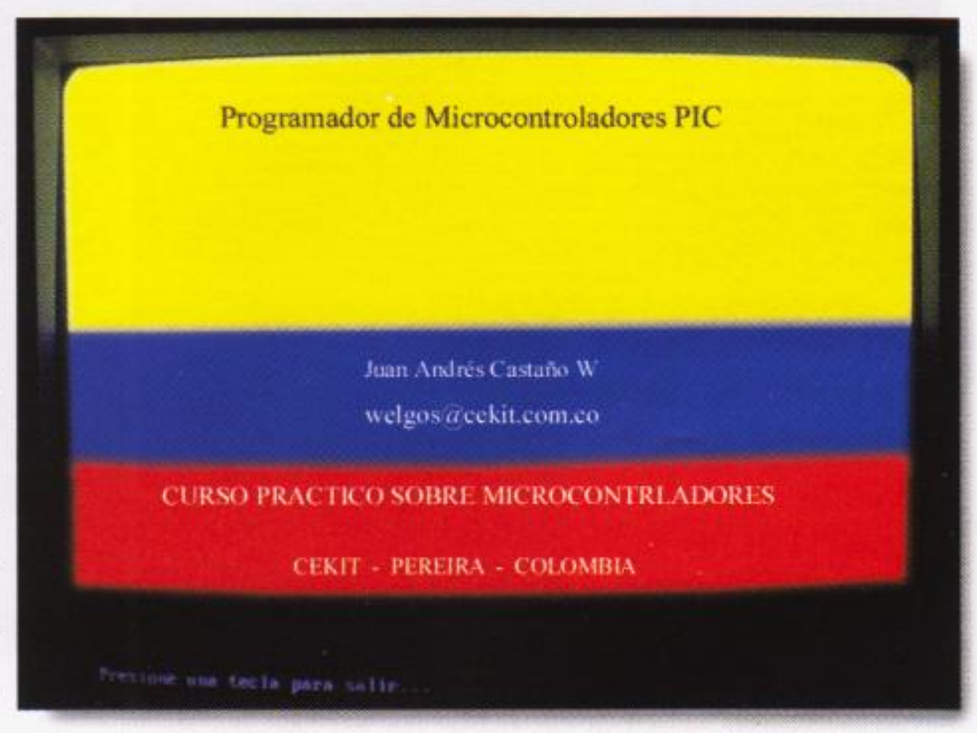


Figura 9.16 *Créditos*

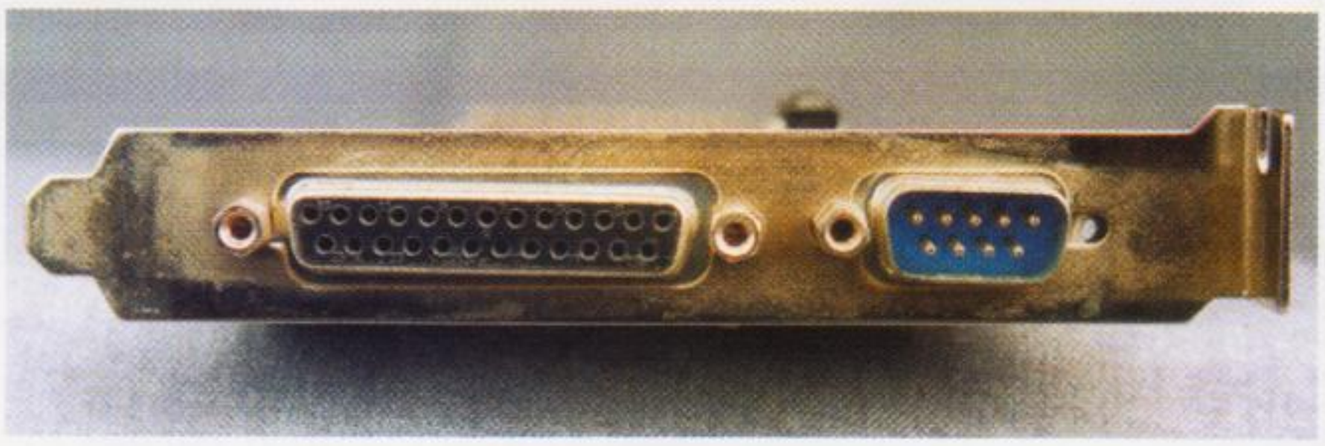


Figura 9.17 *En las computadoras tipo PC es común encontrar en la parte posterior del aparato un conector para los puertos seriales en donde existe un conector DB-9 y un conector DB-25*

ladores con el 16F87x poseen la opción de programación «*in circuit*», ella se caracteriza porque el microcontrolador sólo necesita de dos pines CLOCK y DATA para ser programados. Cuando lleve a cabo programación dentro del circuito, debe tener mucha precaución si va a cambiar las funciones de los puertos, para evitar que se pueda presentar algún cortocircuito que deterioraría la estructura del microcontrolador.

En esta primera versión del programador *in circuit* para microcontroladores PIC, el programa de la computadora se ha desarrollado en Turbo C para DOS. Para instalar adecuadamente el programa en su computadora, debe crear en su disco duro una carpeta con el nombre CEKITP y dentro de ella copiar los dos archivos que vienen en el disquete. Ellos corresponden al nombre de **progra.exe** y **egavga.bgi**. Si estos archivos no se han copiado dentro de una carpeta con el nombre de CEKITP, el programa no funcionará, ya que él está compilado para buscar los archivos en una carpeta ubicada en la unidad C, dentro de una carpeta con el nombre mencionado (C:\CEKITP).

Es posible que su computadora no cuente con un puerto serial del tipo DB-9 disponible para trabajar con la tarjeta, pero que sí cuente con un puerto serial en conector DB-25 (COM2), es decir, que posea un módulo de conexión como el que se muestra en la **figura 9.17**. Este tipo de presentación es muy común en computadoras tipo PC, en donde por lo general el conector DB-9 corresponde al puerto COM1 y el DB-25 corresponde al puerto COM2.

Si ese es el caso, tiene dos opciones: desconectar el ratón de la computadora, conectar allí la tarjeta y configurar la inicialización

COMPUTADORA	
DB - 9	DB - 25
3	2
2	3
7	4
8	5
6	6
5	7
1	8
4	20

Tabla 2. *Conversión serial DB-25 a DB-9*

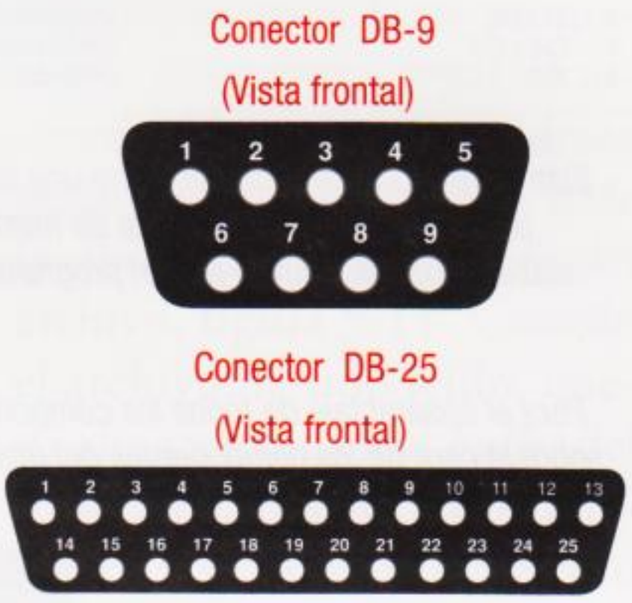


Figura 9.18 *Distribución y nomenclatura de los pines en los conectores DB-9 (arriba) y DB-25 (abajo)*

de la computadora para que sólo trabaje en ambiente DOS, o bien, inicializar Windows normalmente (pero sin conectar el ratón) y ejecutar el programa desde una ventana del DOS. La segunda opción es construir usted mismo el cable de conexión hacia el puerto serial con un conector DB-25; dentro de este kit se han anexado un conector DB-25 y un conector DB-9 adicionales para que usted mismo construya su interfaz de comunicación. Sólo debe hacer un cableado entre los pines de los dos conectores siguiendo la correspondencia indicada en la **tabla 2**. La configuración de los pines de estos dos tipos de conectores la puede apreciar en la **figura 9.18**, con ella puede guiarse fácilmente para construir su propio cable de conexión.

Ensamblaje

LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F84
1	IC2	Driver de comunicación MAX232
1	REG1	Regulador positivo de +12 voltios, 7812
1	REG2	Regulador positivo de +5 voltios, 7805
2	Q1 y Q3	Transistor NPN 2N3904
1	Q2	Transistor PNP 2N3906
1	BR1	Puente rectificador de 1A, W06M
1	D1	Diodo rectificador 1N4148
1	D2	Diodo LED rojo de 5mm
1	D3	Diodo LED verde de 5mm
1	C1	Condensador electrolítico de 1000uF/50V
4	C2,C4,C9 y C12	Condensadores cerámicos de 0.1uF/50V
1	C3	Condensador electrolítico de 47uF/25V
4	C5 a C8	Condensadores electrolíticos de 10uF/25V
2	C10 y C11	Condensadores cerámicos de 22pF/50V
1	XT1	Cristal de cuarzo de 4Mhz
3	R1,R2 y R4	Resistencias de 2.2k-1/4 W
1	R3	Resistencia de 220-1/4W
1	R5	Resistencia de 3.3k-1/4W
2	R6 y R7	Resistencias de 470-1/4W
1	J1	Conector CA/CC
1	J2	Conector DB9 hembra para circuito impreso
1	J3	Conector en línea de 6 pines
1	K-224	Circuito impreso CEKIT
1		Base para circuito integrado de 18 pines
1		Base para circuito integrado de 16 pines
1		Base tipo ZIF ancha de 40 pines
1		Base tipo ZIF ancha de 28 pines
2		Conector DB-9 macho con terminales para soldar
1		Conector DB-9 hembra con terminales para soldar
1		Conector DB-25 hembra con terminales para soldar
3		Metros de cable ribbon de 9 hilos

Este proyecto debe tomarse como una herramienta de laboratorio con la cual puede llevar a cabo un sinnúmero de actividades y proyectos. Por su naturaleza de herramienta de trabajo, nos hemos enfocado en explicar su manejo sin profundizar en la elaboración del firmware y del programa de la computadora. Esperamos que él sea de mucha utilidad en su desempeño por el mundo de los microcontroladores.

Para el ensamblaje de todos los componentes en el circuito impreso del programador, use la guía de montaje que se ha dibujado sobre la cara de los componentes del mismo, tal como se muestra en la figura 9.19, y tenga cuidado con la ubicación de todos los componentes polarizados.

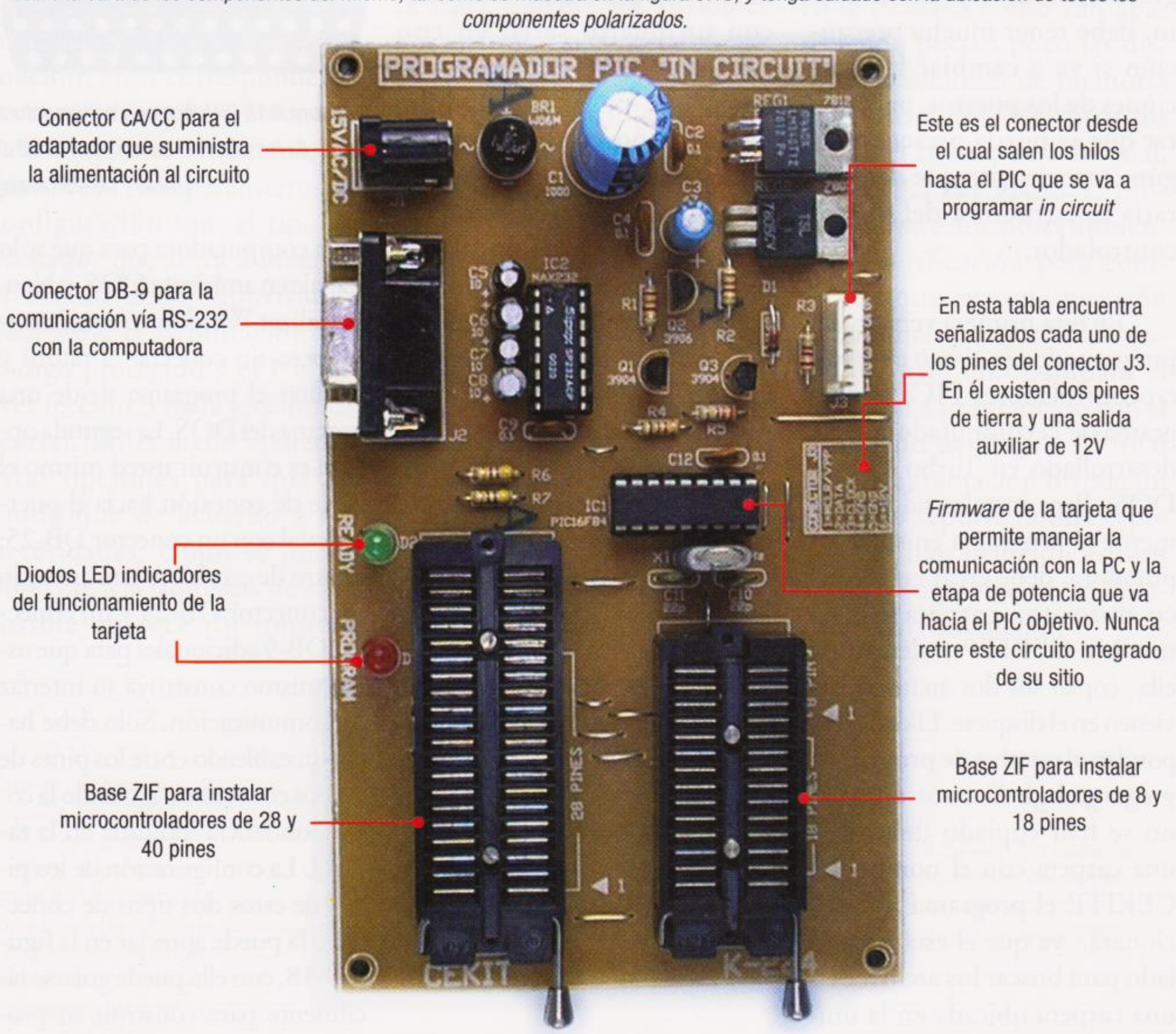
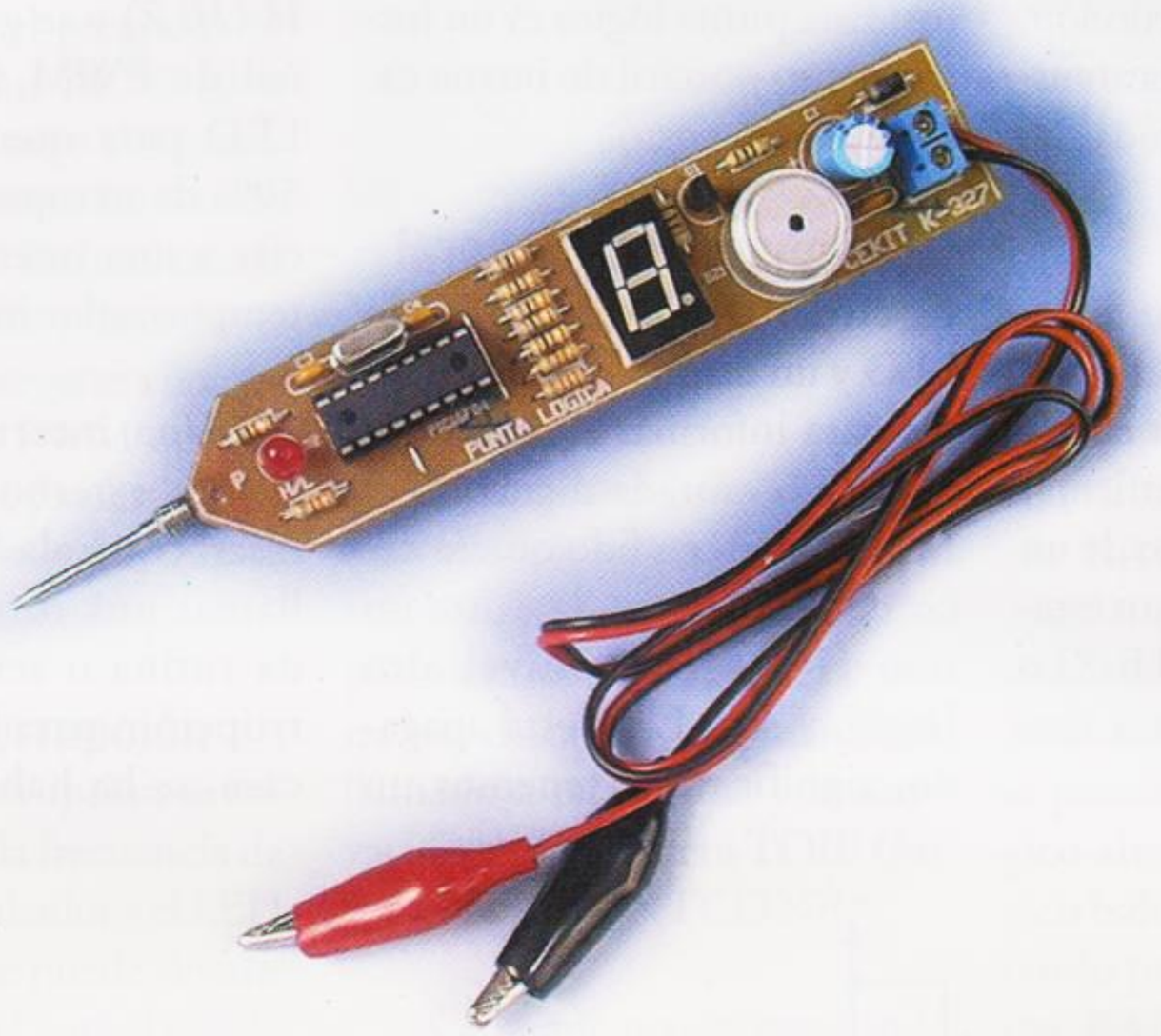


Figura 9.19 *Aspecto final del programador "In Circuit" de microcontroladores PIC*

PROYECTO 10



PUNTA LÓGICA

Simplifique la prueba y la reparación de sistemas digitales construyendo esta práctica punta lógica. Detecta niveles alto, bajo y de alta impedancia; su elemento principal para el proceso es un microcontrolador PIC16F84.

Características

- Punta lógica de tres estados
- Detecta estados de alto (1), bajo (0), alta impedancia (*Hi-Z*) y entrada de pulsos (P)
- Señalización a través de un diodo LED y un *display* de siete segmentos
- Genera una señal PWM para uno de los tipos de señalización
- Tamaño reducido
- Señal audible de cada uno de los estados

Una punta lógica es uno de los elementos más utilizados dentro de un laboratorio de electrónica digital, microprocesadores o microcontroladores. Ella facilita el proceso de construcción, prueba, reparación, mantenimiento e instalación de toda clase de sistemas digitales.

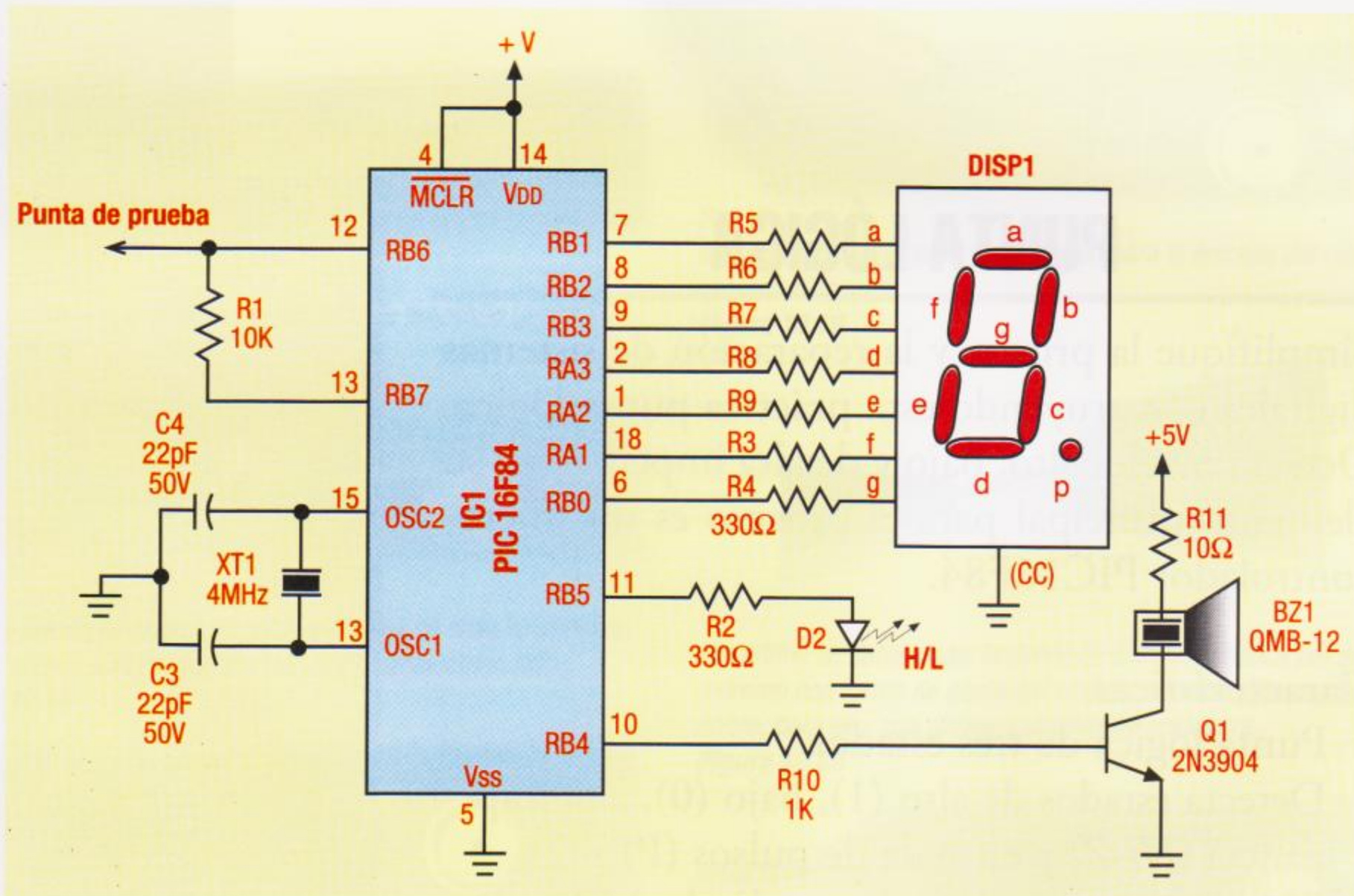
Utilizando una punta lógica se puede determinar fácilmente el estado de una entrada o salida digital y confirmar la presencia en ese punto de un nivel lógico alto o bajo, un estado de alta impedancia (Hi-Z) o una entrada de pulsos a una alta frecuencia.

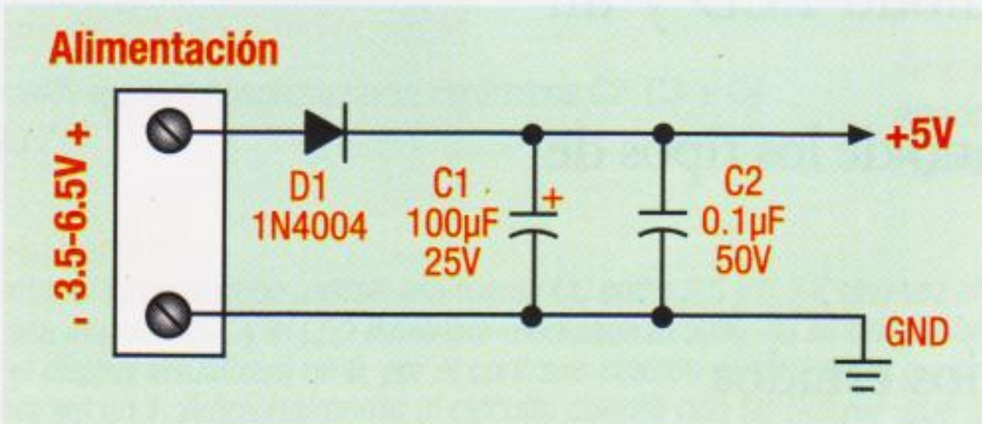
Las puntas lógicas para uso profesional son instrumentos relativamente costosos y, por la misma razón, no están siempre accesibles a todos los presupuestos. Esta punta lógica es un instrumento portátil de buena calidad y bajo costo.

Funcionamiento general

El circuito utiliza un diodo LED y un display de 7 segmentos para informar el estado lógico de la entrada digital. Si el LED está encendido quiere decir que en la entrada existe un uno (1) lógico o nivel alto (*high*), y si el LED está apagado, significa que tenemos un cero (0) lógico o nivel bajo (*low*) en la entrada. Si en la entrada existe un estado de alta impedancia, se mostrará en un *display* de siete segmentos la letra H (*Hi Z*) y se generará una señal de PWM sobre el diodo LED para que se ilumine al 50% de su capacidad; esto gracias a una interrupción por el temporizador interno (*timer0*).

Una interrupción es un evento externo que obliga al microcontrolador a invocar o llamar una rutina denominada rutina o servicio de interrupción, para lo cual, en este caso, se ha habilitado al tem-





ESTADO	DISPLAY	LED
1	1	On
0	0	Off
HZ	H	PWM (50%)
Pulsos	P	Oscila

Figura 10.1 *Diagrama esquemático de la punta lógica*

porizador interno del microcontrolador (*timer0*) para que cuente los pulsos internos de reloj de la máquina y genere una interrupción cada vez que este registro se llene y provoque un sobreflujo.

Cuando se ha detectado un estado lógico alto o bajo, se enviará por el pin RB4 (pin 10) una señal en una frecuencia determinada (el nivel lógico bajo tendrá una frecuencia más baja) para que el usuario tenga una señal audible del nivel que está detectando. Adicionalmente se ha dispuesto un *display* de siete segmentos en el cual se puede observar una letra o un número por cada estado que se presente. Cuando hay un nivel lógico bajo se visualizará un cero (**0**), cuando es nivel alto se visualizará un uno (**1**), cuando se encuentre en estado de alta impedancia se visualizará una **H**, y cuando entre una señal de pulsos se presentará una **P**.

Diagrama esquemático

El diagrama esquemático de este proyecto lo puede apreciar en la **figura 10.1**. Consta de una entrada de alimentación, un diodo LED indicador de estado, un transductor piezoeléctrico miniatura (*buzzer*), un *display* de siete segmentos y un microcontrolador PIC 16F84.

La entrada de alimentación está conformada por unos bornes de conexión (conector de entrada) destinados a recibir los cables de alimentación, el diodo rectificador de protección D1, y los condensadores C1 y C2 que se comportan como filtros y cuya función es estabilizar la tensión de alimentación aplicada al circuito y limitar la entrada de señales de alta frecuencia hasta el microcontrolador.

Para garantizar la ejecución continua del programa por parte del microcontrolador y el buen funcionamiento de todos sus circuitos internos de apoyo (temporizadores, *buffers*, registros, contadores y circuitos de *reset*, entre otros), es necesario que la alimentación que se le suministre, esté dentro de los limites permitidos por su tecnología de fabricación en cuanto a su magnitud CC y CA (rizado), los cuales establecen que los microcontroladores PIC pueden ser alimentados por voltajes que van entre 3 y 6VCC y con un factor de rizado inferior o igual al 10% de su valor CC.

Para suministrar la energía al circuito de la punta lógica, se usa la propia fuente de alimentación del circuito bajo prueba, y para ello se usa un cable de alimentación con dos caimanes miniatura en uno de sus extremos, que se conecta a los bornes de entrada del circuito impreso. Para evitar que se induzcan señales eléctricas ruidosas y que la magnitud de la tensión de alimentación sea atenuada, éste cable debe mantenerse con una longitud inferior a 1m.

Por las características de la tecnología de fabricación del PIC y debido a la caída de tensión ocasionada por el diodo de protección D1 en polaridad directa, el voltaje de alimentación de la punta lógica debe estar entre 3.5 y 6.5V, es decir, que puede utilizarse para probar el estado lógico de circuitos digitales TTL y CMOS con bajo nivel de alimentación.

El diodo rectificador de propósito general D1 que abre o cierra el paso de la corriente hacia el circuito, dependiendo de la polaridad entre sus terminales (ánodo y cátodo), se ha incluido en la entrada como una protección para el circuito

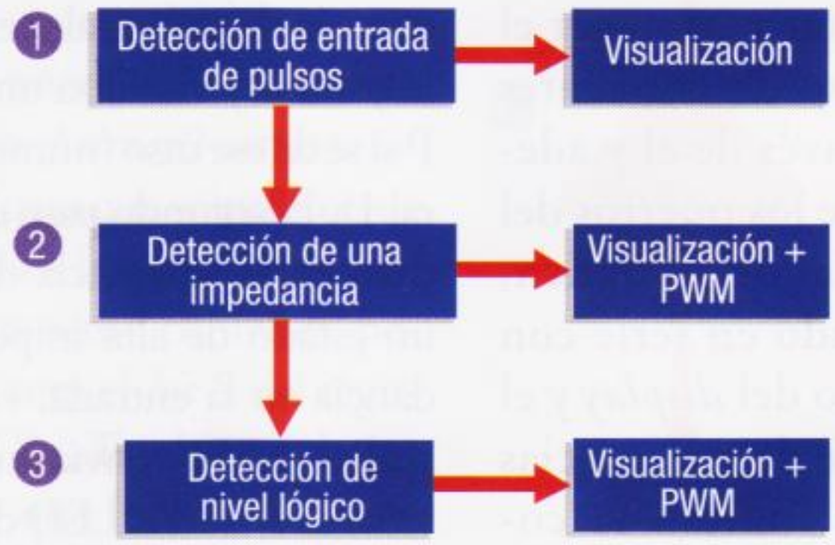


Figura 10.2 *Diagrama de bloques del programa*

```
MAIN:
        BSF     ESTADO, 5       ; CONMUTA A LA PAGINA 1
A       MOVLW   B'01000000'     ; SE CONFUGRA EL FUNCIONAMIENTO DE LOS PINES
        MOVWF   TRISB           ; DE ACUERDO A LO DISPUESTO EN EL DIAGRAMA
        MOVLW   B'00000000'     ; ESQUEMATICO. SOLO HAY UNA ENTRADA, EL
        MOVWF   TRISA           ; PIN NUMERO SEIS DEL PUERTO B.
        BCF     OPCION, 5       ; CONFIGURO EL TIMER COMO TEMPORIZADOR
        BCF     INTCON, 2       ; BORRO BANDERA DE INT. DEL TMR0
        BSF     INTCON, 7       ; HABILITADOR GLOBAL DE INT.
        BCF     ESTADO, 5       ; VUELVO A LA PAGINA 0
        CLRF    BANDERA

loop    NOP                     ; EL PRIMER PASO ES DETERMINAR SI EXISTE
PULSO   BTFSS   PORTB,6         ; EN LA ENTRADA UNA FRECUENCIA DE PULSOS
        GOTO    PULSO_BAJO      ; EL METODO ES MUY SENCILLO, SOLO SE LEE
PULSO_ALTO                      ; LA ENTRADA RB6, Y SE VERIFICA QUE ESE
        CALL    RETARDO         ; NIVEL PERMANEZCA DURANTE CIERTO PERIODO
        BTFSC   PORTB,6         ; DE TIEMPO, EL CUAL ES DETERMINADO POR
        GOTO    loop1           ; LA RUTINA RETARDO. CAMBIANDO EL VALOR DE
        GOTO    EST_PULSO       ; ESA RUTINA, CAMBIAMOS LA FRECUENCIA DE
                                ; DETECCION.
PULSO_BAJO                      ; EN ESTA LINEA ESTAMOS LEYENDO DE NUEVO RB6
        CALL    RETARDO         ; DESPUES DE UN RETARDO PARA CONSTATAR QUE SIGUE
        BTFSS   PORTB,6         ; EXISTIENDO UN NIVEL LÓGICO BAJO.
        GOTO    loop1           ; SI AUN EXISTE SIGNIFICA QUE NO HAY
                                ; ENTRADA DE PULSOS
EST_PULSO                       ; Y PODEMOS CONTINUAR EL PROGRAMA.
        BSF     PORTB,1         ; SI SE HA DETECTADO UNA ENTRADA DE PULSOS,
        BSF     PORTA,1         ; ACTIVAMOS LOS PINES DE LOS PUERTOS
        BSF     PORTA,2         ; CORRESPONDIENTES PARA QUE SE VISUALICE LA LETRA P
        BSF     PORTB,0         ; PARA INDICAR QUE ESTA ENTRANDO A LA
B                               ; PUNTA LOGICA
        BSF     PORTB,2         ; UNA SERIE DE PULSOS A UNA FRECUENCIA
                                ; DETERMINADA.
        BCF     PORTB,3
        BCF     PORTA,3         ; CUANDO SE VISUALIZA LA P, REGRESAMOS A
                                ; LA ETIQUETA
        BCF     PORTB,5         ; loop PARA OBSERVAR EN QUE MOMENTO NOS
                                ; SALIMOS DE
        BCF     PORTB,4         ; ESTE ESTADO.
        GOTO    loop
```

Figura 10.3 *Código que detecta la existencia de entrada de pulsos*

de la punta lógica; su propósito es evitar los daños permanentes que se pueden ocasionar en el PIC como consecuencia de una inversión prolongada en la polaridad del voltaje de alimentación.

Para limitar y proteger el diodo del paso de corrientes excesivas a través de él y además evitar que los puertos del microcontrolador se dañen, se han instalado en serie con cada segmento del *display* y el LED una serie de resistencias (R2-R9) que limitan la corriente a menos de 10mA por cada uno.

Descripción del programa

En la **figura 10.2** puede observar el diagrama de bloques general con las funciones que hace el dispositivo. Inicialmente detecta si existe una frecuencia de pulsos alta en la entrada y visualiza una P si se da ese caso (numeral 1). El segundo paso es detectar la existencia de un estado de alta impedancia en la entrada, visualiza una H y envía una señal de PWM al LED de salida (numeral 2). Si no ha ocurrido alguno de los dos eventos anteriores se pasa al tercer paso, en el cual se detecta qué nivel lógico existe en la punta de prueba y envía una señal en frecuencia para activar el zumbador y tener así una indicación audible de la medición.

El primer paso es verificar si en la entrada existe una señal de pulsos, el código de esta función se observa en la **figura 10.3** y su diagrama de flujo en la **figura 10.4**. Para resolver este problema se pensó en una solución muy sencilla: se lee la entrada, se espera un tiempo determinado y de nuevo se vuelve a leer para verificar si el nivel leído inicialmente permanece allí o ha cambiado. Si el nivel ha cambiado se puede concluir que es una señal que se mueve a una frecuencia considerable, pero si el nivel leído permanece se puede decir que se está leyendo un nivel que no cambia tan

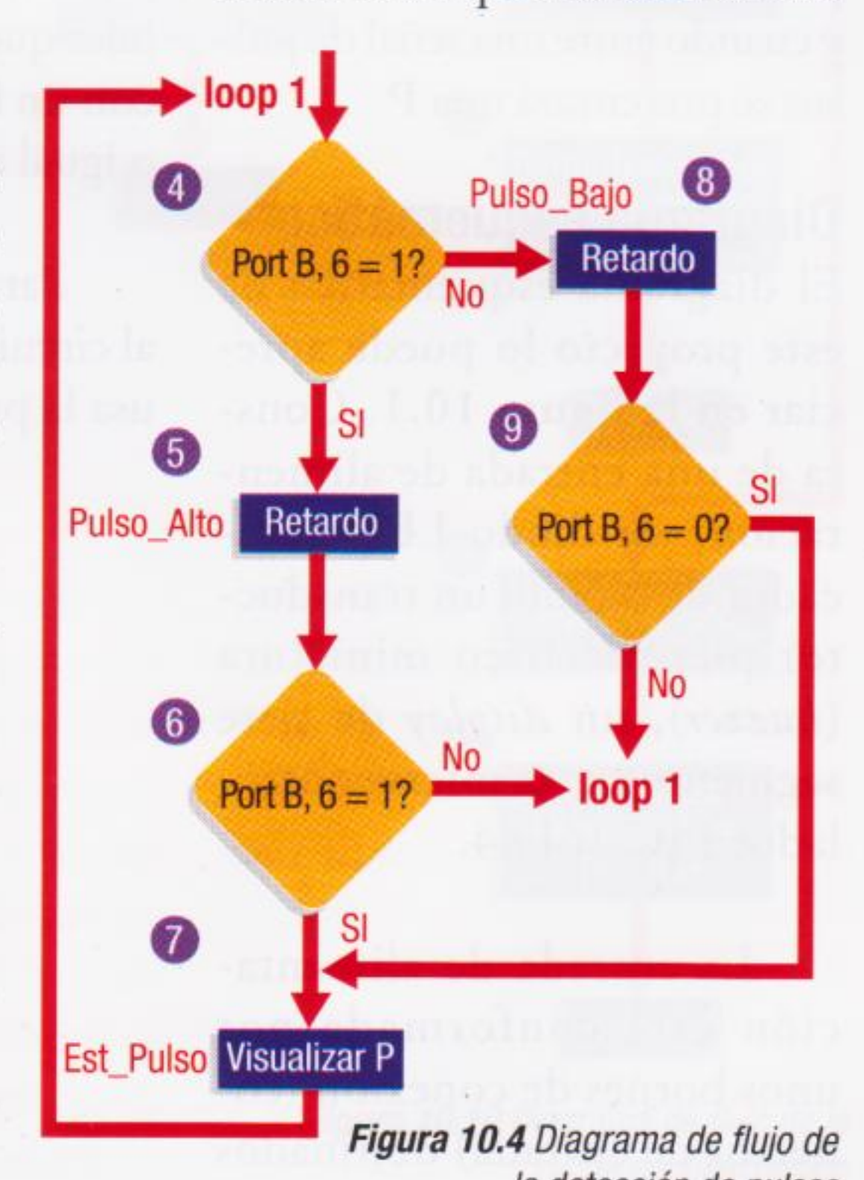


Figura 10.4 *Diagrama de flujo de la detección de pulsos*

```
loop1  BCF    PORTB, PHZ    ; RB7 AHORA ESTA EN BAJO...
       BTFSC  PORTB, PPRUEB ; PRUEBO SI RB6 ES IGUAL A RB7
       GOTO   nohz
       BSF    PORTB, PHZ    ; .. Y LUEGO PONGO EN ALTO
       BTFSS  PORTB, PPRUEB ; PRUEBO SI RB6 ES IGUAL A RB7
       GOTO   nohz
hz     BSF    BANDERA,1     ; SI LA ENTRADA ES HI Z
       BSF    PORTB,0       ; SE ACTIVA EL BIT 1 DEL REGISTRO
C      BCF    PORTB,1       ; BANDERA PARA PODER RECONOCER
       BSF    PORTB,2       ; ESTE ESTADO DENTRO DE LA INTERRUPCION
       BSF    PORTB,3       ; Y MANDAR LA FRECUENCIA ADECUADA.
       BSF    PORTA,1       ; AL MISMO TIEMPO SE ACTIVA LA LETRA H
       BSF    PORTA,2       ; EN EL DISPLAY DE SIETE SEGMENTOS.
       BCF    PORTA,3       ; FINALMENTE ACTIVO LA INTERRUPCION:
       BCF    ESTADO, 5     ; VOY A LA PAGINA 1
       BSF    INTCON, 5     ; HABILITO INT. POR TMR0
       BCF    ESTADO, 5     ; VUELVO A LA PAGINA 0
       GOTO   loop

nohz   BCF    BANDERA,1     ; SI LA ENTRADA NO ESTA EN ESTADO DE ALTA Z
       BCF    PORTB, PHZ    ; PONGO EL PIN RB7 EN BAJO, PARA PRUEBAS
D                            ; POSTERIORES
       BTFSS  PORTB, 6      ; Y LEO FINALMENTE EL PIN 6 DEL PUERTO B
       GOTO   BAJO           ; Y VOY A LA RUTINA CORRESPONDIENTE (ALTO O BAJO)

ALTO   BSF    PORTB, PLED   ; SI LA ENTRADA ESTA EN ALTO, LED
                             ; ENCENDIDO FULL
       BSF    BANDERA,0     ; ACTIVO EL BIT 0 DEL REGISTRO BANDERA
       BSF    PORTB,2       ; PARA RECONOCER ESTE ESTADO DENTRO DE LA
       BSF    PORTB,3       ; INTERRUPCION.
E      BCF    PORTB,1       ; LUEGO HABILITO LOS PINES ADECUADOS QUE
                             ; PERMITIRAN
       BCF    PORTB,0       ; VISUALIZAR EL NUMERO 1 EN EL DISPLAY
       BCF    PORTA,1       ; DE SIETE SEGMENTOS.
       BCF    PORTA,2       ; FINALMENTE HABILITO EL FUNCIONAMIENTO
       BCF    PORTA,3       ; DE LA INTERRUPCION POR EL TMR0
       BCF    ESTADO, 5     ; VOY A LA PAGINA 1
       BSF    INTCON, 5     ; HABILITO INT. POR TMR0
       BCF    ESTADO, 5     ; VUELVO A LA PAGINA 0
       GOTO   loop

BAJO   BCF    PORTB, PLED   ; SI LA ENTRADA ESTA EN BAJO, SE DEBE APAGAR
       BCF    BANDERA,0     ; EL LED DE SALIDA, PONER EN CERO EL BIT 0
       BSF    PORTB,1       ; DEL REGISTRO BANDERA PARA SER DETECTADO
F      BSF    PORTB,2       ; ESTE ESTADO DENTRO DE LA INTERRUPCION,
       BSF    PORTB,3       ; Y ACTIVAR LOS PINES CORRESPONDIENTES DE
       BSF    PORTA,1       ; LOS PUERTOS PARA VISUALIZAR EL NUEMRO
       BSF    PORTA,2       ; O EN EL DISPLAY DE SIETE SEGMENTOS.
       BSF    PORTA,3       ; FINALMENTE SE ACTIVA LA INTERRUPCION
       BCF    PORTB,0       ; POR TIMER0.
       BCF    ESTADO, 5     ; VOY A LA PAGINA 1
       BSF    INTCON, 5     ; HABILITO INT. POR TMR0
       BCF    ESTADO, 5     ; VUELVO A LA PAGINA 0
       GOTO   loop
```

Figura 10.5 *Detección y operación ante un estado de alta impedancia*

rápidamente. En los bloques 4, 6 y 9 del diagrama de flujo y parte B del programa, se hace esta operación. Observe que el tiempo de espera está dado por una rutina de tiempo denominada **RETARDO** y que si ha encontrado este estado, debe visualizar en el *display* de siete segmentos la letra **P**.

En la porción A del código se configura el *timer0* como temporizador (para que se incremente con los pulsos internos de la máquina) y no como contador, poniendo en cero el *bit* 5 del registro OPCION. También se borra la bandera que indica una interrupción debido al *timer0* (lo cual es obligación del programador) y se activa el *bit* habilitador global de interrupciones (GIE).

Ahora el sistema debe determinar si en la punta de prueba existe una condición de alta impedancia. Para determinar la existencia de este estado siga un procedimiento muy sencillo: remítase al código de la **figura 10.5** y al diagrama de flujo de la **figura 10.6**. El pin 7 del puerto B está conectado a través de una resistencia de 100K a la entrada de prueba (RB6); si en el pin RB7 se coloca un cero (bloque 10 del diagrama de flujo, parte C del programa) e inmediatamente se lee el pin RB6 (bloque 11) y allí se detecta cero, es posible que hayan ocurrido dos cosas: la primera es que la punta de prueba en ese momento esté leyendo un cero lógico el cual no se afecta con el cero proveniente

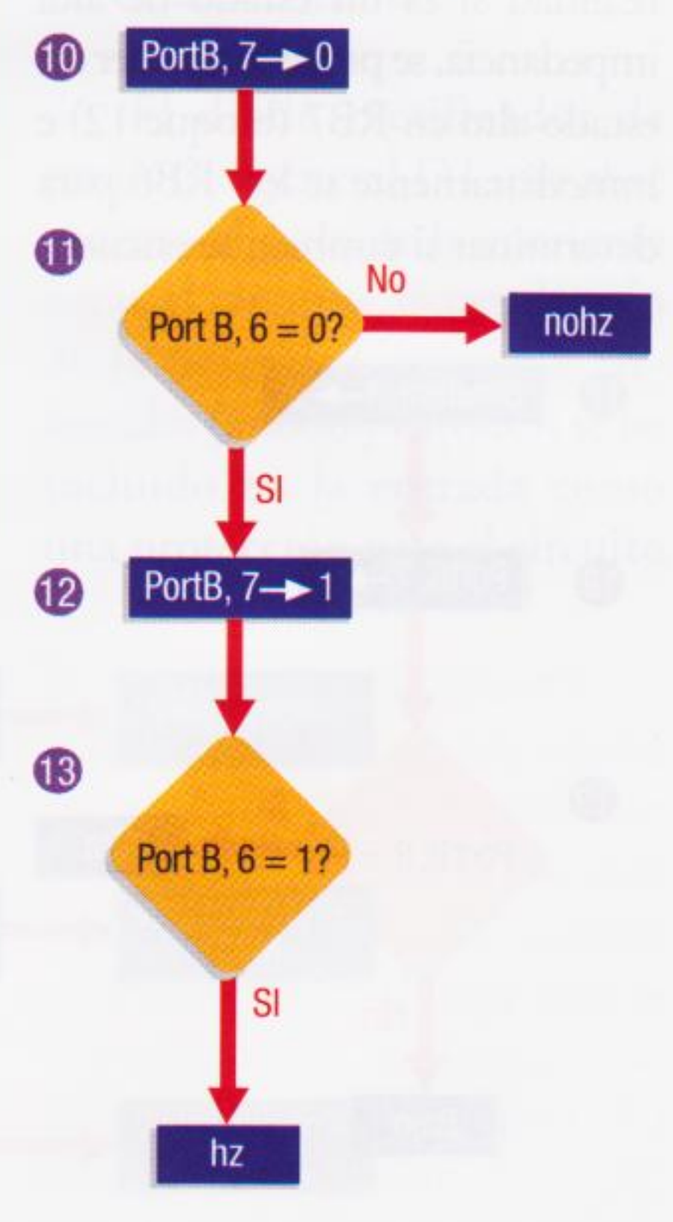


Figura 10.6 *Diagrama de flujo de la detección de alta impedancia*

Figura 10.7 *Operación cuando se detecta alta impedancia*

del RB0; la segunda, es que está en estado de alta impedancia y en realidad el cero que se lee en el puerto RB6 sea exactamente el mismo que envió por RB7; es decir, que la señal externa no afectó para nada la lectura del PIC. Observe que si en este preciso momento la punta lee una señal lógica alta, no habrá ningún problema de cortocircuito gracias a R1.

Para estar seguro de que en realidad sí es un estado de alta impedancia, se procede a poner un estado alto en RB7 (bloque 12) e inmediatamente se lee RB6 para determinar si también se encuentra en alto (bloque 13). Si las dos pruebas son positivas, entonces se puede concluir que tiene un estado de alta impedancia, **figura 10.7**.

```
ORG 0x000            ;vector de reset
GOTO   MAIN
ORG 0x004            ; vector de interrupcion
BTFSS  BANDERA,1     ;CHEQUEO BANDERA DE QUE SE HA
GOTO   UNOCERO       ; PRESENTADO UN ESTADO DE ALTA
                     ; IMPEDANCIA.
BTFSS  PORTB, 5      ; SI ESE ES EL CASO, CADA QUE
BSF    PORTB, 5      ; OCURRA UNA INTERRUPCION, SE ORIGINA
BTFSS  PORTB, 4      ; POR LOS PINES 4 Y 5 DEL PUERTO B,
BSF    PORTB, 4      ; UNA SEÑAL DE PWM, GENERADA CAMBIANDO
BTFSC  PORTB, 5      ; ALTERNADAMENTE EL ESTADO DE ESTAS
BCF    PORTB, 5      ; DOS LINEAS, LAS CUALES CORRESPONDEN
BTFSC  PORTB, 4      ; AL DIODO LED INDICADOR Y AL ZUMBADOR.
BCF    PORTB, 4      ;
BSF    ESTADO, 5     ; VOY A LA PAGINA 1
BCF    INTCON, 2     ; BØRRO LA BANDERA DE INTER. DEL TMRØ
BCF    ESTADO, 5     ; VUELVO A LA PAGINA Ø
MOVLW  Øx99          ; CONFIGURO EL PORCENTAJE DE CICLO UTIL
MOVWF  TMRØ          ; EN EL TEMPORIZADOR TMRØ.
RETFIE
```

Figura 10.9 *Código de la interrupción para el estado de alta impedancia*

En ese momento se pone un uno en el *bit* 1 del registro BANDERA (bloque 14), se ponen en los puertos los *bits* necesarios para visualizar la letra **H** y se habilita al *timer0* para que genere su interrupción (bloque 16), esto se hace en el programa al activar el *bit* 2 del registro INTCON.

Si alguna de las dos condiciones no se cumple, puede esperar un estado lógico normal, **figura 10.8**, se pone en cero el *bit* 1 del registro BANDERA (bloque 17) y se coloca de nuevo RB7 en cero para que el sistema funcione normalmente.

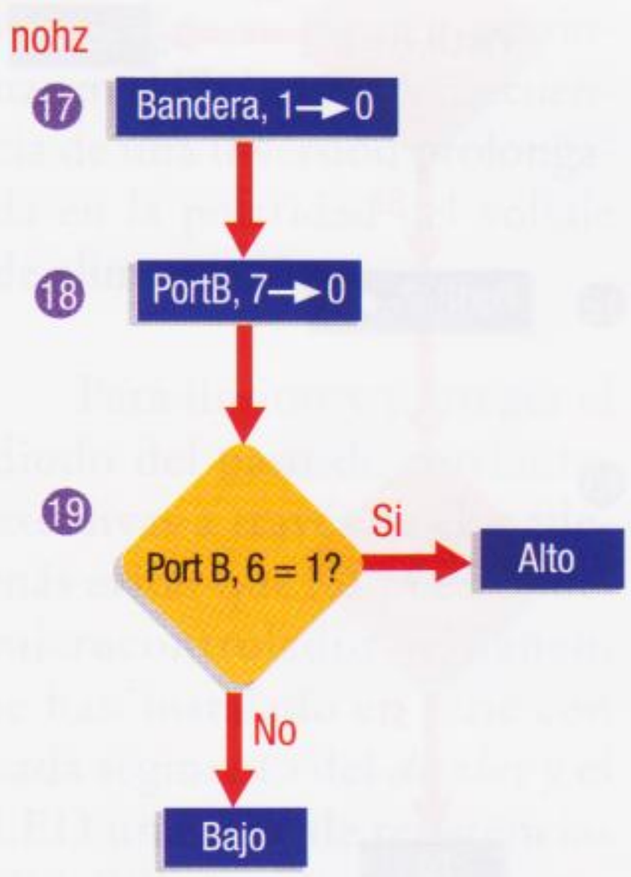


Figura 10.8 *Operación cuando no se detecta estado de alta impedancia*

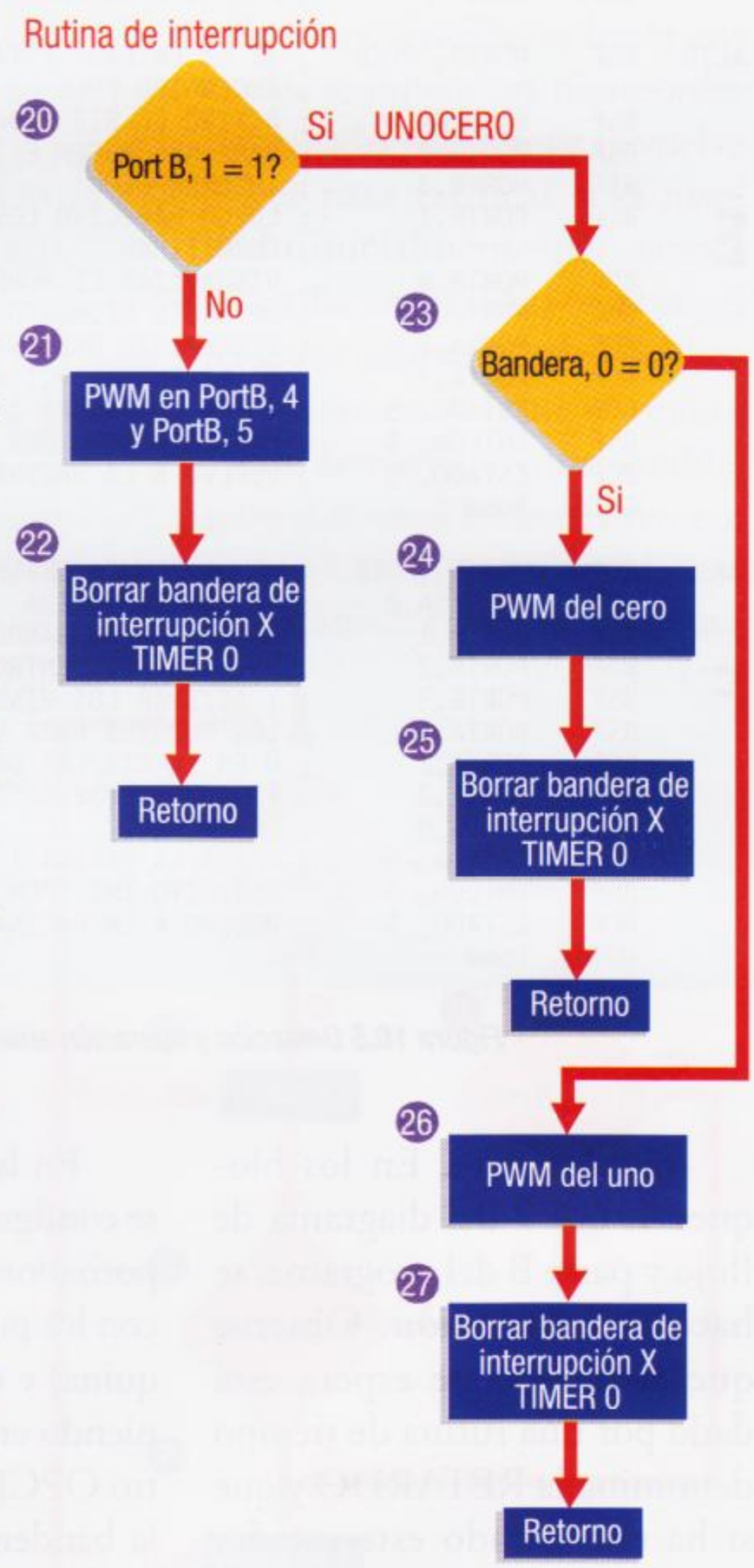


Figura 10.10 *Diagrama de flujo de la rutina de interrupción*

Ahora puede evaluar el estado de la punta de prueba (bloque 18) preguntando por el estado del pin RB6, esto se hace en la sección D del programa; según el estado de ese pin, el programa lo conduce a la rutina **ALTO** (si la entrada está en alto) o a la rutina **BAJO** si lo que detectó fue un cero lógico, bloque 19. En cada una de esas rutinas se activa o desactiva el LED indicador y se genera una señal en frecuencia PWM que acciona el zumbador. Finalmente regresa a la rutina **loop** para que se repita el proceso completo.

```
UNOCERO BTFSC  BANDERA,0     ; SI NO HAY ESTADO DE ALTA IMPEDANCIA,
        GOTO   UNO           ; SE VERIFICA EL BIT 0 DEL REGISTRO
                             ; BANDERA
        BTFSS  PORTB, 4      ; PARA DETERMINAR SI SE HA LEIDO UN CERO
        BSF    PORTB, 4      ; O UN UNO. SI SE HA LEIDO UN CERO (BIT
        BTFSC  PORTB, 4      ; CERO -> 0), SE MANDA UNA SEÑAL DE PWM
                             ; HACIA
        BCF    PORTB, 4      ; EL ZUMBADOR A UNA FRECUENCIA MENOR
                             ; QUE LA DE HZ.
        BSF    ESTADO, 5     ; VOY A LA PAGINA 1
        BCF    INTCON, 2     ; BØRRO LA BANDERA DE INTER. DEL TMRØ
        BCF    ESTADO, 5     ; VUELVO A LA PAGINA Ø
        MOVLW  Ø×71          ; CONFIGURO UN PORCENTAJE DE CICLO UTIL
        MOVWF  TMRØ          ; EN EL TEMPORIZADOR TMRØ
        RETFIE
```

Figura 10.11 *Interrupción para el estado lógico bajo*

Rutina de interrupción

La rutina de interrupción se encarga de generar la frecuencia de salida para el zumbador y el LED indicador. Ella se puede dividir en tres partes: una para el estado lógico alto, otra para el estado bajo y otra para el estado de alta impedancia. Cada una de estas partes debe generar una frecuencia distinta. La porción para el estado de alta impedancia la puede apreciar en el código de la **figura 10.9** y en el diagrama de flujo de la **figura 10.10** se encuentra el esquema general de esta rutina. La primera operación que debe hacer es preguntar por el estado del *bit* 1 del registro **BANDERA** (bloque 20) para determinar si se ha entrado a esta rutina después de haber ocurrido un estado de alta impedancia (recuerde que cuando se detectó el estado de alta impedancia este *bit* se puso en uno).

Cuando se ha detectado un estado de alta impedancia se activa el *timer0* para que cuente los pulsos de reloj internos de la máquina; después de un tiempo determinado, el *timer0* se llena y se desborda, originando el llamado a esta rutina y la activación de la bandera TOIF (*bit 2* del registro INTCON).

Cada que ocurre este llenado el programa invoca esta rutina en la cual se cambia el estado de la salida en los pines 4 y 5 del puerto B, y se limpia la bandera TOIF (bloque 22) para que la rutina de interrupción pueda ser invocada de nuevo. Debido a que la interrupción se da tan rápidamente esta rutina se invoca a una rata muy alta, de esta manera se origina una señal cuadrada llamada PWM. Si el *bit* 1 del registro **BANDERA** no está en uno, el programa lo lleva a la etiqueta **UNOCERO** en donde se detecta si la interrupción se provocó para un estado lógico alto o para un estado lógico bajo. Esto se hace preguntando por el *bit* cero del registro **BANDERA**, el cual fue configurado previamente para cada estado antes de activar la interrupción (bloque 23).

Para cada uno de los dos casos debe enviar un PWM a distintas frecuencias, borrar la bandera TOIF y luego retornar al programa, **bloques 23-26** del diagrama de flujo y **figuras 11** y **12**.

```
UNO     BSF    PORTB, PLED   ;
        BTFSS  PORTB, 5      ; SI SE HA DETERMINADO UN ESTADO LOGICO ALTO,
        BSF    PORTB, 5      ; SE MANDA UNA NUEVA SEÑAL DE PWM HACIA EL
        BTFSS  PORTB, 4      ; ZUMBADOR, A UNA NUEVA FRECUENCIA MAYOR A
        BSF    PORTB, 4      ; LA ENVIADA PARA EL ESTADO DE ALTA
                             ; IMPEDANCIA.
        BTFSC  PORTB, 5      ; NOTE QUE ESTA RUTINA DE INTERRUPCION SE
        BCF    PORTB, 5      ; HA DIVIDIDO EN TRES SUBRUTINAS, CADA UNA DE LAS
        BTFSC  PORTB, 4      ; CUALES SE ENCARGA DE ENVIAR UNA FRECUENCIA
                             ; DISTINTA
        BCF    PORTB, 4      ; PARTIENDO DE UN SOLO TEMPORIZADOR.
        BSF    ESTADO, 5     ; VOY A LA PAGINA 1
        BCF    INTCON, 2     ; BØRRO LA BANDERA DE INTER. DEL TMRØ
        BCF    ESTADO, 5     ; VUELVO A LA PAGINA Ø
        MOVLW  ØxFØ          ; CONFIGURO EL PORCENTAJE DE CICLO UTIL
        MOVWF  TMRØ          ; EN EL TEMPORIZADOR TMRØ
        RETFIE
```

Figura 10.12 *Interrupción para el estado lógico alto*

Ensamblaje del proyecto

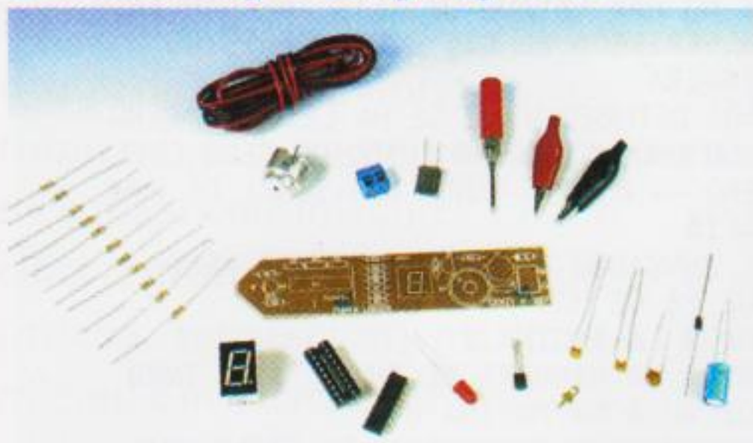
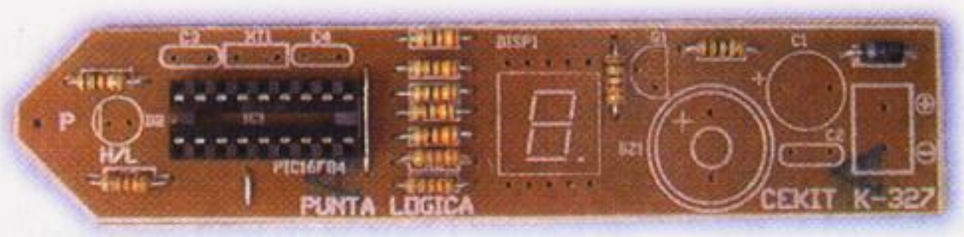


Figura 10.13 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar la punta lógica de este proyecto.*

LISTA DE MATERIALES		
Cant.	**Ref.**	**Descripción**
1	IC1	Microcontrolador PIC16F84
1	Q1	Transistor de propósito general NPN, 2N3904
1	DISP1	*Display* de siete segmentos cátodo común
1	BZ1	*Buzzer* piezoeléctrico miniatura, QMB-12
1	D1	Diodo rectificador de propósito general 1N4004
1	D2	Diodo LED rojo de 5mm
1	C1	Condensador electrolítico de 100uF - 25V
1	C2	Condensador cerámico de 0.1uF - 50V
2	C3 y C4	Condensadores cerámicos de 22pF - 50V
1	XT1	Cristal de quarzo de 4MHz
1	R1	Resistencia de 10k - 1/4W
8	R2 a R9	Resistencias de 330 - 1/4W
1	R10	Resistencia de 1k - 1/4W
1	R11	Resistencia de 10 - 1/4W
1	K-327	Circuito impreso CEKIT
1		Conector de tornillo de 2 pines
1		Base para circuito integrado de 18 pines
1		Punta de prueba metálica
1		Caimán miniatura, rojo
1		Caimán miniatura, negro
1m		Cable duplex No 24, rojo y negro

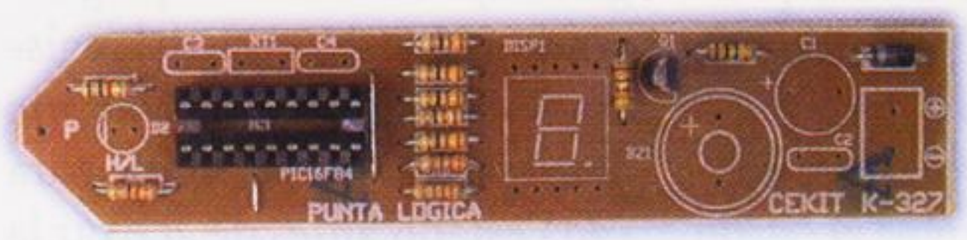
Este proyecto constituye una valiosa herramienta para la reparación y desarrollo de proyectos electrónicos que debe tenerse en el laboratorio de electrónica de cualquier persona. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:



Paso 1

Con un alambre delgado, instale y suelde los puentes marcados en el circuito impreso, instale y suelde también las resistencias, la base y el diodo rectificador.

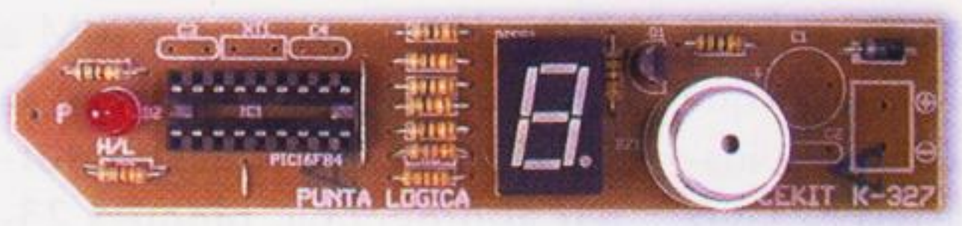
Figura 10.14



Paso 2

Luego instale y suelde el transistor NPN de propósito general Q1 .

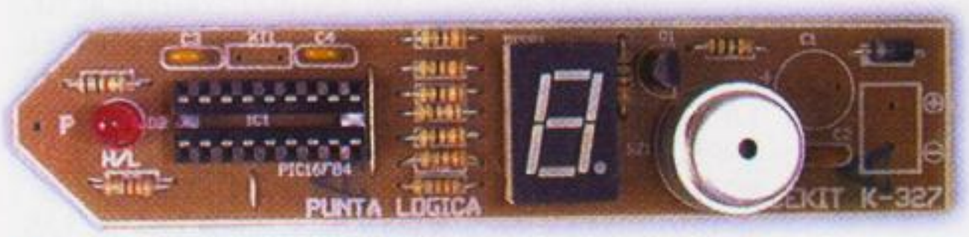
Figura 10.15



Paso 3

Instale y suelde el diodo LED (D2), el display de siete segmentos cátodo común (DISP1) y el transductor piezoeléctrico (BZ1).

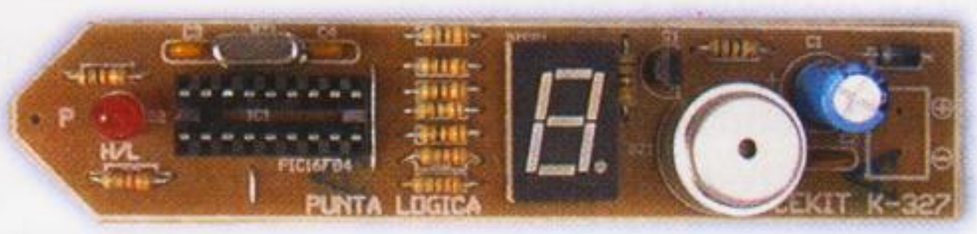
Figura 10.16



Paso 4

Instale y suelde los tres condensadores cerámicos C2, C3 y C4.

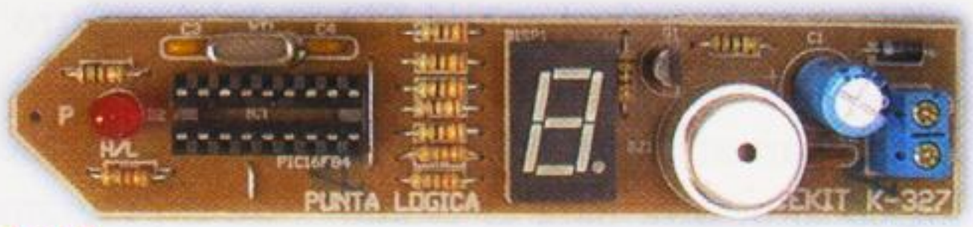
Figura 10.17



Paso 5

Instale y suelde el condensador electrolítico (C2) y el cristal de cuarzo de 4 MHz (XT1).

Figura 10.18



Paso 6

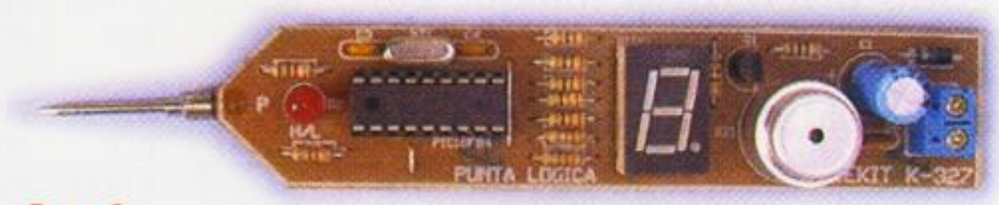
Instale y suelde el borne de conexión para la entrada del cable de alimentación.

Figura 10.19

Paso 7

Por último se debe soldar la punta de prueba metálica por el extremo con forma de punta del circuito impreso y por su lado de cobre, para ello, use pinzas y abundante soldadura.

Figura 10.20



Paso 8

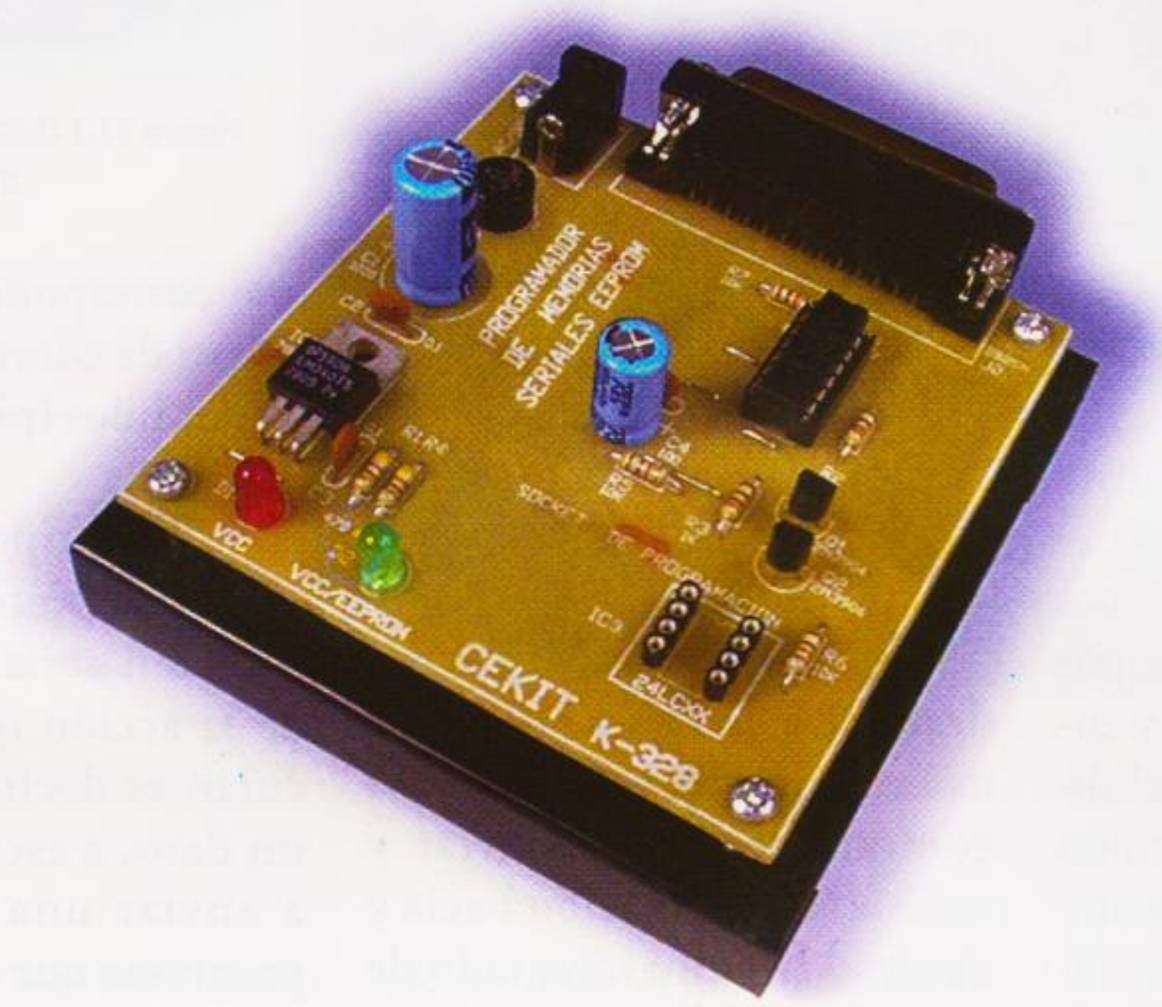
Apariencia final de la punta lógica completamente ensamblada.

Figura 10.21

Paso 9. Pruebas finales

*Para alimentar el circuito debe usarse una fuente CC entre 3.5 y 6.5 V, una vez alimentado debe visualizarse la letra **H** en el display indicando el estado de alta impedancia, y el LED iluminará modulado al 50% de su brillo total. Cuando la punta sea llevada a un nivel lógico bajo, el LED se apagará y el display visualizará un **0**, por el contrario cuando el nivel de su punta sea alto (más de 2.5V), el LED se iluminará plenamente y el display mostrará un **1**. Adicionalmente el circuito cuenta con un Buzzer que indica con dos tonos los estados alto y bajo.*

PROYECTO 11



PROGRAMADOR DE MEMORIAS SERIALES

Las memorias EEPROM de tipo serial han tenido gran acogida en el mercado electrónico gracias a su reducido tamaño físico y a su bajo consumo de potencia, convirtiéndose en dispositivos ideales para proyectos que requieran almacenamiento de algún tipo de información. En este proyecto usted podrá construir su propio programador de memorias seriales con el cual podrá grabar datos mediante una computadora.

Características

- Tamaño reducido
- Programa de usuario interactivo y fácil de manejar
- Soporta memorias 24XX
- Conexión al puerto paralelo de la computadora

Las memorias de tipo EEPROM tienen como su principal cualidad permitir el almacenamiento y la sobreescritura de datos por medio de los voltajes de operación normal de los circuitos electrónicos y sostener la información por muchos años sin energía de alimentación. Se pueden encontrar circuitos integrados de memorias EEPROM paralelas, compatibles pin a pin con circuitos de memoria RAM o de memoria EPROM. Este tipo de memorias, precisamente por ser de interfaz paralela, tiene muchos pines externos por medio de los cuales recibe y entrega los datos y permite el direccionamiento de las distintas posiciones de almacenamiento. Debido a esto, los circuitos integrados son de gran tamaño físico, impidiendo ser utilizados en aplicaciones que requieran tamaño reducido.

Con las memorias EEPROM de interfaz serial, el control se ha reducido solamente a unos cuantos pines que son utilizados para entrada o salida de datos en forma serial (1 ó 2 pines), habilitación (1 pin), reloj de sincronización (1 pin), direccionamiento de dispositivo (3 pines) que no existe en las de interfaz paralela y, por último, los pines de alimentación del circuito (2 pines). Los datos y la dirección de las posiciones de memoria usan únicamente uno o dos pines dependiendo del tipo de comunicación utilizada (dos o tres hilos). La velocidad de transferencia de datos puede variar desde 100 KHz hasta 10 MHz, dependiendo del tipo de memoria y del sistema de comunicación utilizados.

El programador que se describe a continuación hace uso del puerto paralelo de la computadora para enviar y recibir la información hacia y desde el circuito integrado de memoria EEPROM de tipo serial que se encuentre insertado en el circuito externo. Todo el conjunto, circuito y programa, puede adquirirse bajo la referencia K-328 de CEKIT.

Descripción de la memorias seriales EEPROM

En la **figura 11.1** puede apreciar el diagrama de pines de la memoria con referencia 24LC04

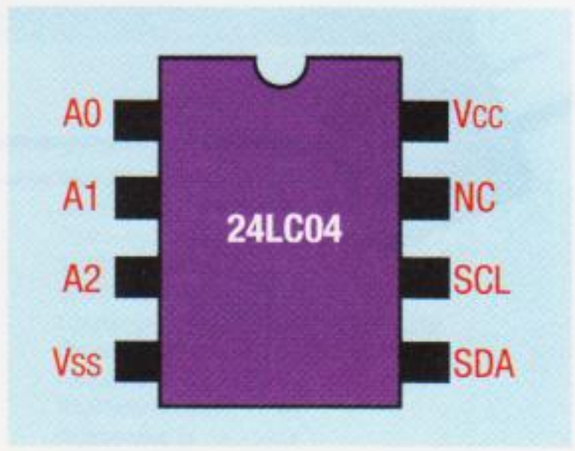


Figura 11.1 *Distribución de pines de la memoria serial 24LC04*

que corresponde a los dispositivos de comunicación serial de dos hilos (pin SDA y SCL).

Este tipo de memorias utiliza un protocolo de comunicación serial que depende de la acción que se va a ejecutar, es decir, si se va a leer un dato, a escribir, o si se va a enviar una dirección. El programa que ejecuta la computadora deberá seguir paso a paso la secuencia del protocolo, dependiendo del tipo de memoria que se desea programar o leer.

Si la memoria tiene más de 256 posiciones, el direccionamiento se hace por medio de páginas siendo las primeras 256 la página 0 (cero), las posiciones 256 a 511 la página 1 y así sucesivamente.

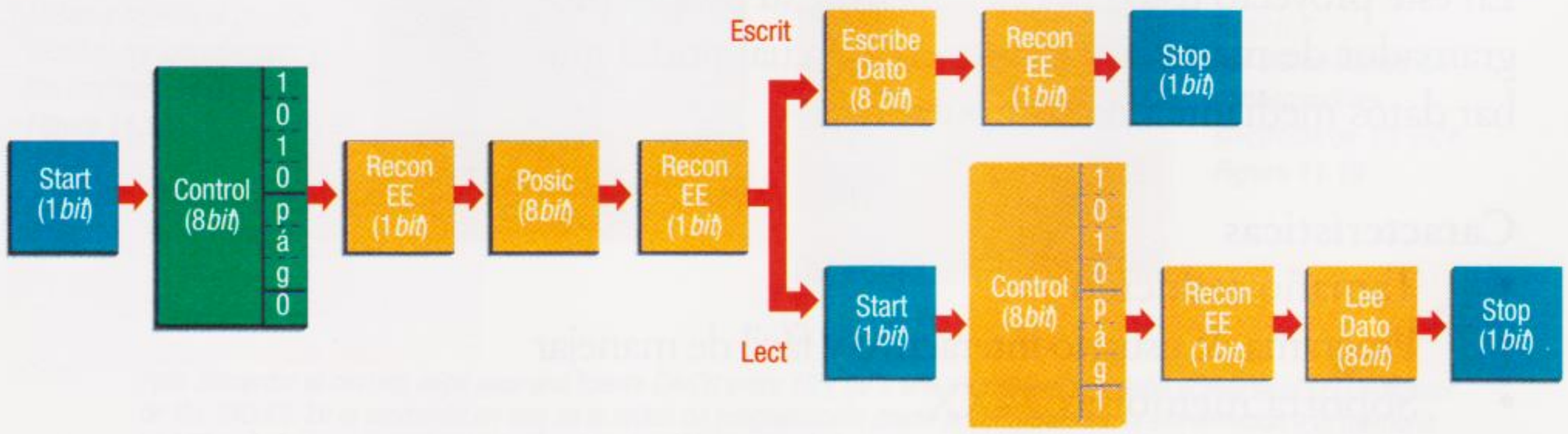


Figura 11.2 *Secuencia del proceso de lectura y escritura en memorias tipo 24XX*

Dentro de la información que se debe enviar a las memorias seriales se incluyen uno o varios *bits* correspondientes a los números de página de la memoria.

Secuencia de operación de las memorias

En la **figura 11.2** puede apreciar el diagrama de bloques indicando la secuencia que debe seguir para escribir o para leer en memoria de tipo 24LCXX. Observe que para cada operación debe recibir un reconocimiento de la memoria que consiste en un pulso bajo durante un ciclo del reloj.

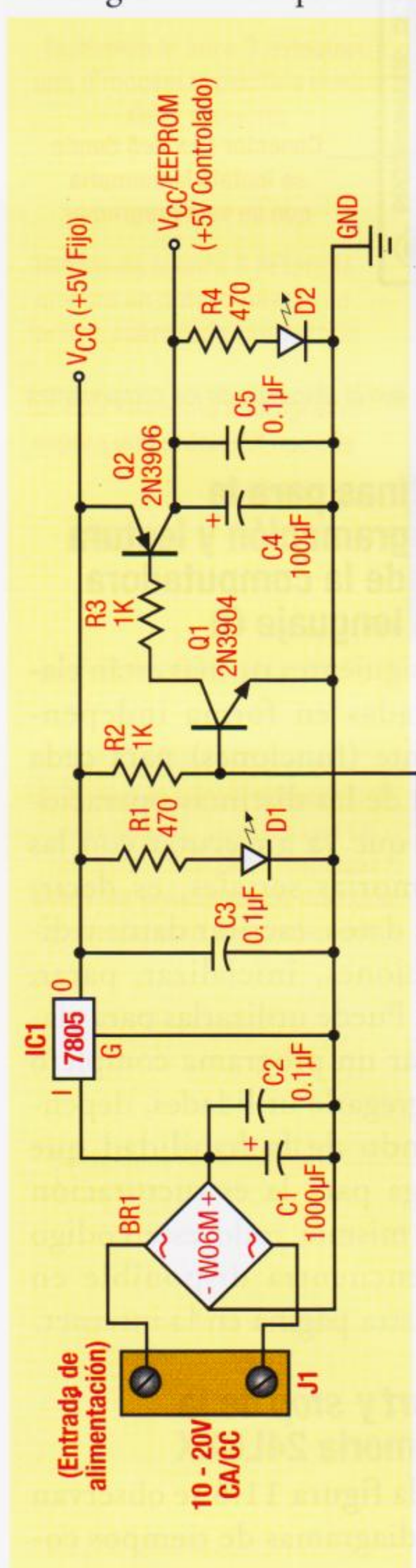
Durante este reconocimiento la computadora debe tener una salida alta en el pin de datos, con el fin de permitir a la memoria escribir el pulso bajo y poder saber por medio de éste si realmente está entendiendo la información que le está enviando.

Descripción del circuito

Para la comunicación entre la computadora y la memoria, debe elaborar un pequeño circuito electrónico que tiene como fin servir de interfaz entre dichos dispositivos y alojar los circuitos integrados que se van a programar. En la **figura 11.3** se muestra el diagrama esquemático y en la **figura 11.4** el circuito impreso con la respectiva ubicación de los componentes sobre el mismo.

A través del puerto paralelo de la computadora, el mismo que utiliza la impresora, puede leer los datos que contiene la memoria y mostrarlos en la pantalla con su respectiva dirección. Así mismo, puede enviar un dato a determinada dirección de los circuitos de memoria para que éstas lo almacenen en forma definitiva hasta que se sobreescriba o borre tal posición.

Con el fin de poder insertar o retirar los *chips* de memoria del circuito electrónico sin necesidad de apagar manualmente la fuente de alimentación, se ha diseñado un pequeño sistema de tal forma que las memorias estén energizadas (+Vcc) solamente durante su programación. A través de una de las compuertas de IC2, se envía la señal para que los transistores TR1 y TR2 suministren la alimentación positiva (Vcc/EEPROM) a los circuitos de memoria. Por medio del LED2 puede visualizar el instante en que las memorias están con alimentación positiva



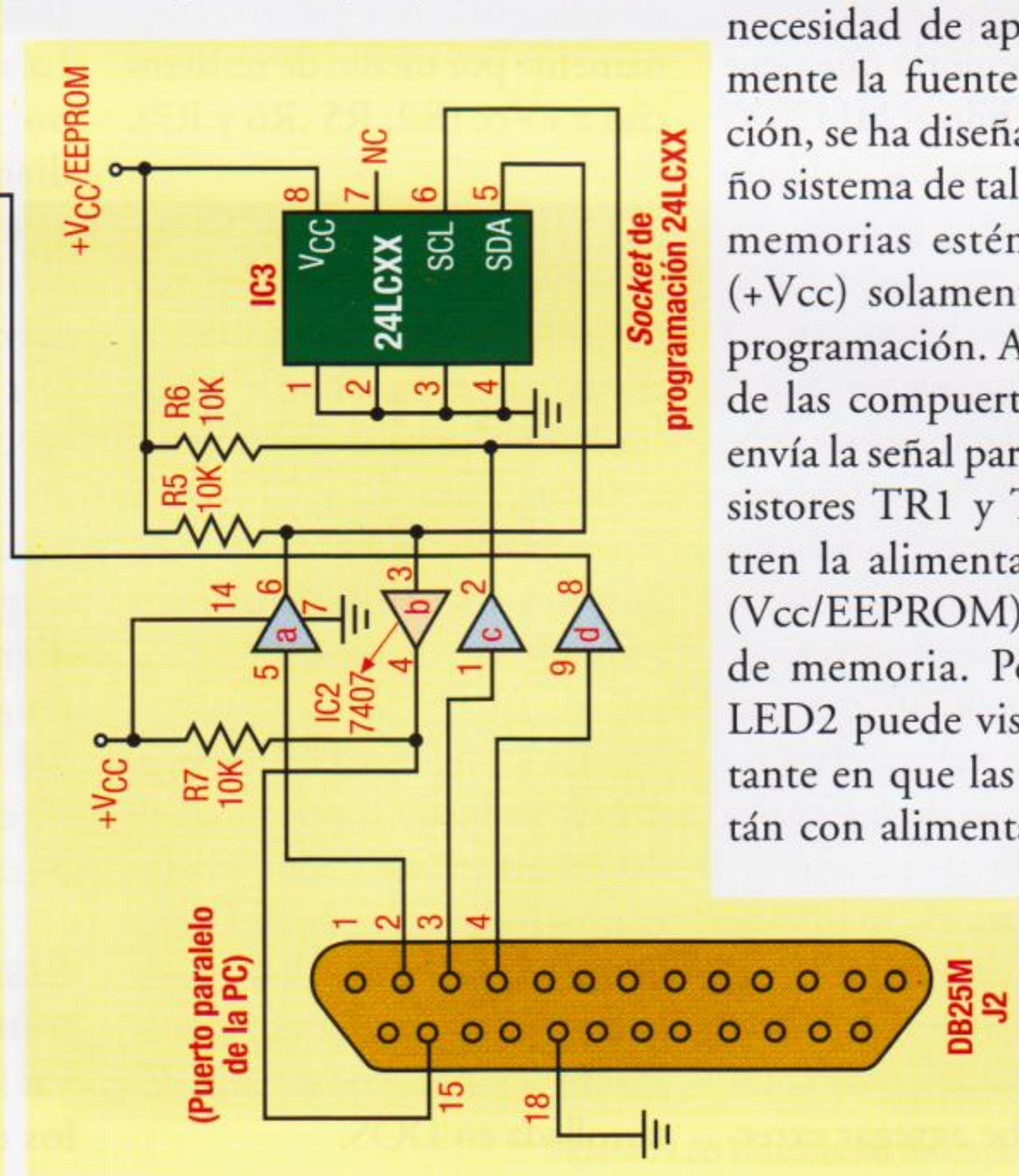


Figura 11.3 *Diagrama esquemático del programador de memorias seriales*

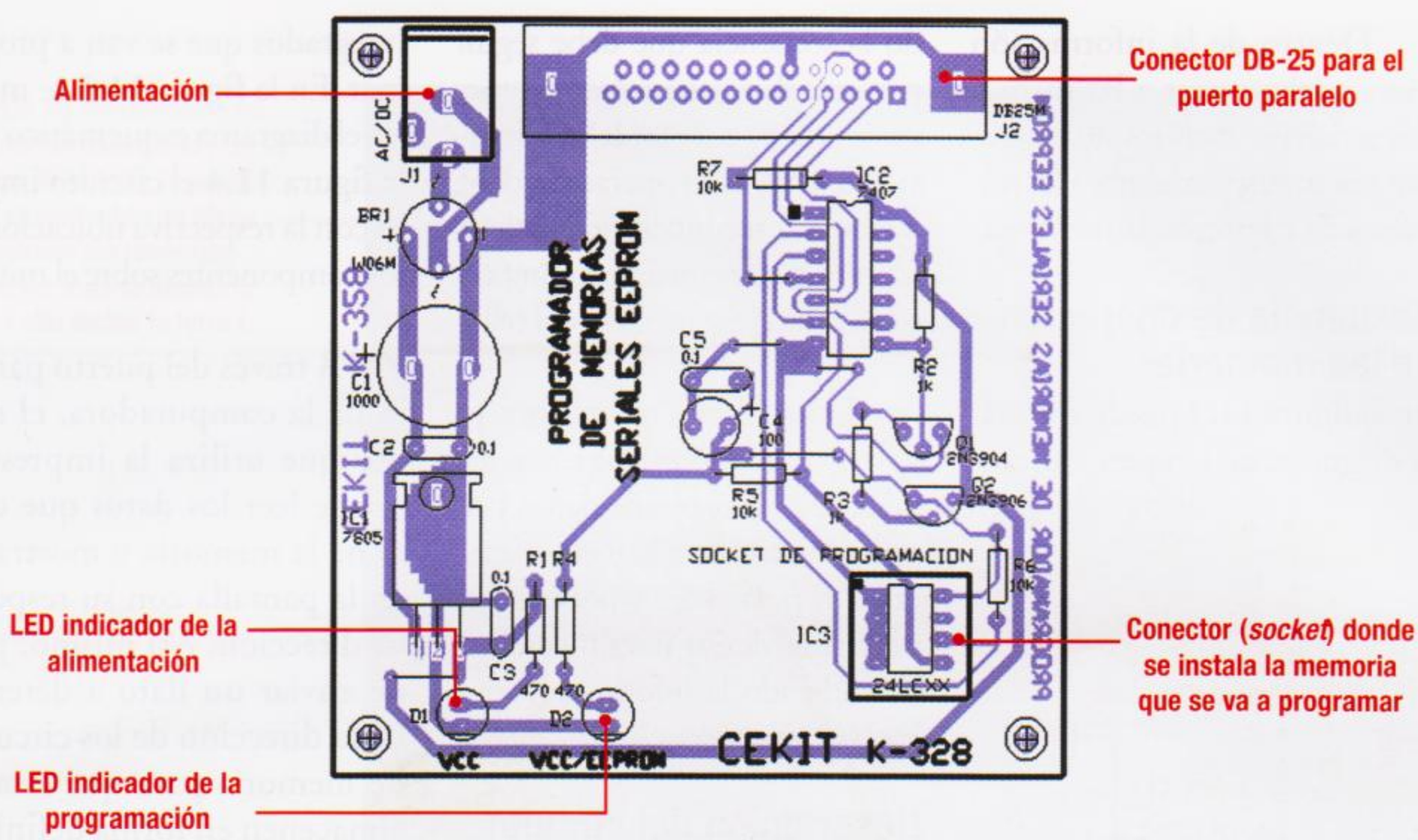


Figura 11.4 *Circuito impreso con la ubicación de los componentes*

y con el LED1 el estado de los 5 voltios de alimentación general del circuito.

Con la memoria que tiene interfaz a dos hilos (24LCXX) se debe tener especial cuidado ya que utiliza un mismo pin para entrada y salida de datos. En este caso se deben utilizar compuertas de colector abierto de tal modo que, si hay conflicto entre los datos que entran y salen, no ocurra un cortocircuito y posible daño de los componentes electrónicos. Se ha utilizado el circuito integrado IC2 con 6 compuertas de colector abierto, de las cuales hacemos uso de tres para la memoria y una para el circuito de control de alimentación (Vcc/EEPROM) mencionado anteriormente. A este tipo de compuertas les debe agregar externamente una resistencia *pull up* ya que, como son de colector abierto, el estado lógico alto de salida debe ser puesto externamente por medio de resistencias a +Vcc (R2, R5 ,R6 y R7).

El puente de diodos BR1 tiene como objeto rectificar el voltaje de entrada de alimentación al circuito, el cual puede ser de corriente alterna o de corriente continua, entre 10 y 20 voltios. El integrado regulador IC1 suministra +5 voltios continuos constantes al circuito.

En las **figuras 11.5 (a, b, c, d y e)** se ilustra la presentación del programa de la computadora con indicación de cada una de sus partes. Con ellas puede guiarse para manejar adecuadamente la interfaz de usuario desarrollada en DOS.

Rutinas para la programación y lectura desde la computadora (en lenguaje C)

La siguientes rutinas están elaboradas en forma independiente (funciones) para cada una de las distintas operaciones que va a ejecutar con las memorias seriales, es decir: leer datos, escribir datos y direcciones, inicializar, parar, etc. Puede utilizarlas para elaborar un programa completo y agregarle utilidades, dependiendo de la habilidad que tenga para la estructuración del mismo; todo este código se encuentra disponible en nuestra página en la internet.

Start y *stop* de la memoria 24LCXX

En la **figura 11.6** se observan los diagramas de tiempos co-

rrespondientes al ciclo de inicialización y de finalización que se deben efectuar sobre la memoria tipo 24LCXX, antes y después de cualquier operación sobre ella.

Reconocimiento en las memorias 24XX

Las memorias de tipo 24XX envían un reconocimiento a la computadora a través del pin de datos indicando que la operación en uso ha terminado en forma correcta. Si la memoria estaba recibiendo información, debe enviar un nivel a través de la línea de datos para permitirle a la me-

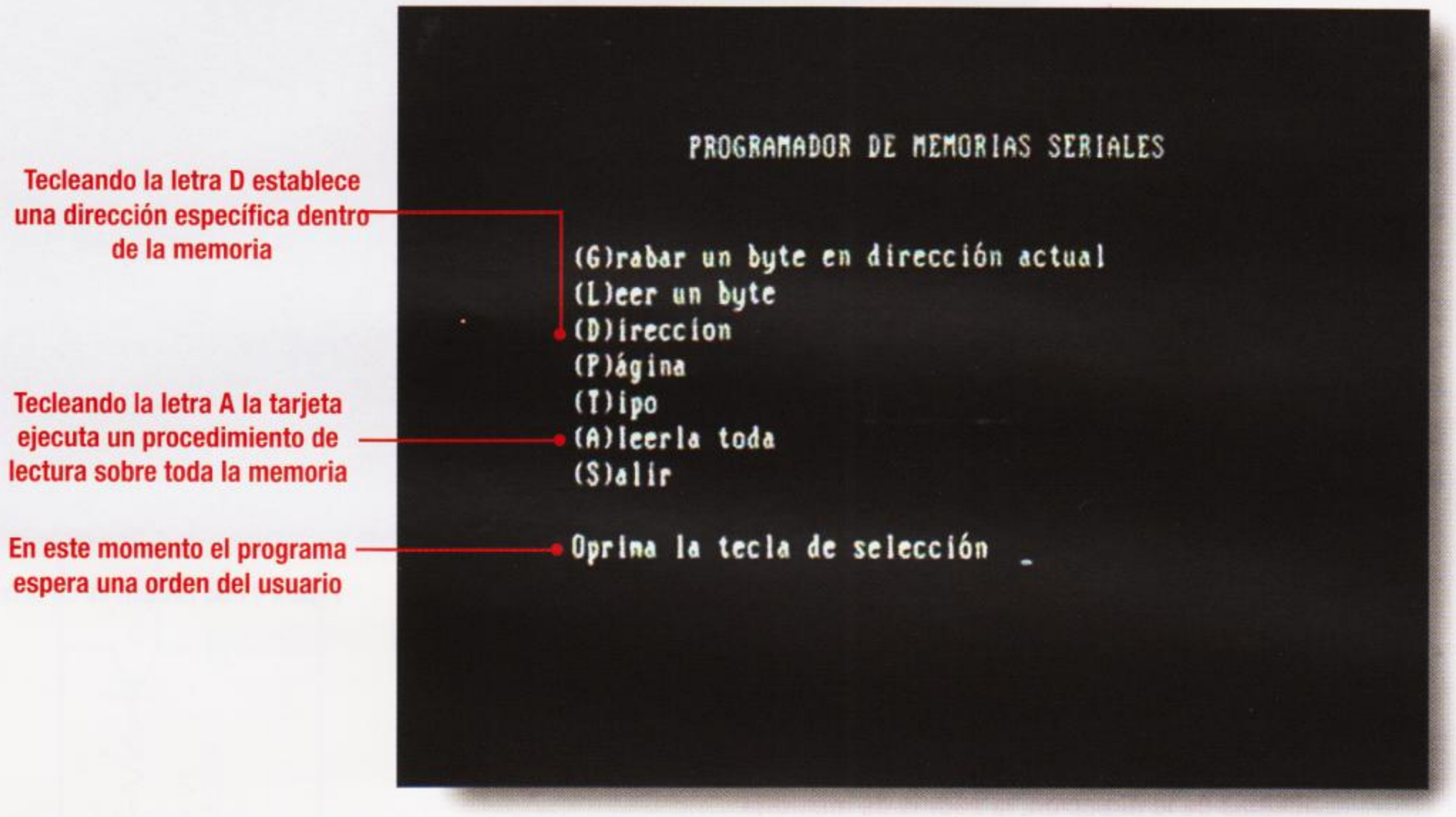


Figura 11.5a Menú principal del programa

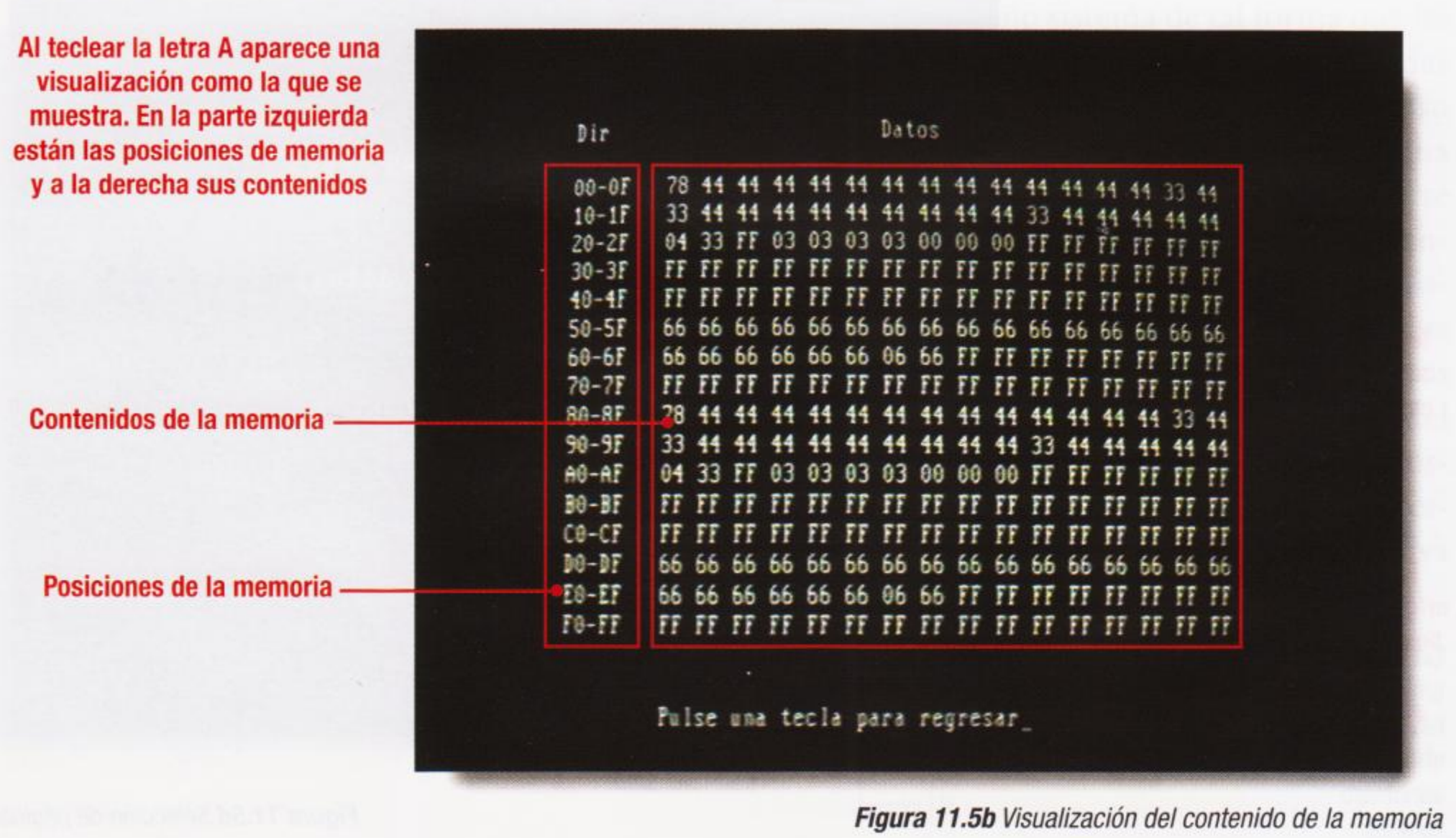


Figura 11.5b Visualización del contenido de la memoria

Cuando ha establecido una dirección específica por medio de la letra D, puede leer el contenido de dicha posición sin tener que observar toda la memoria. Para ello teclee la letra L

Tecleando la letra S sale del programa

```
(G)rabar un byte en dirección actual
(L)eer un byte
(D)ireccion
(P)ágina
(T)ipo
(A)leerla toda
(S)alir

Oprima la tecla de selección

Entre tipo de memoria
(1) 24LCXX
```

Figura 11.5c *Lectura de un byte*

El procedimiento para el manejo de este programa es establecer primero una dirección inicial donde va a programar un dato. A partir de allí el programa le pedirá que llene las direcciones consecutivas de la memoria. También es posible seleccionar, además de la dirección, la página de memoria.

Teclendo la letra P puede acceder a alguna página específica de la memoria

En este momento se origina en la pantalla una línea de comandos en la cual usted debe seleccionar la página

```
PROGRAMADOR DE MEMORIAS SERIALES

(G)rabar un byte en dirección actual
(L)eer un byte
(D)ireccion
(P)ágina
(T)ipo
(A)leerla toda
(S)alir

Oprima la tecla de selección
Entre # de página (1 ó 2): 1_
```

Figura 11.5d *Selección de página*

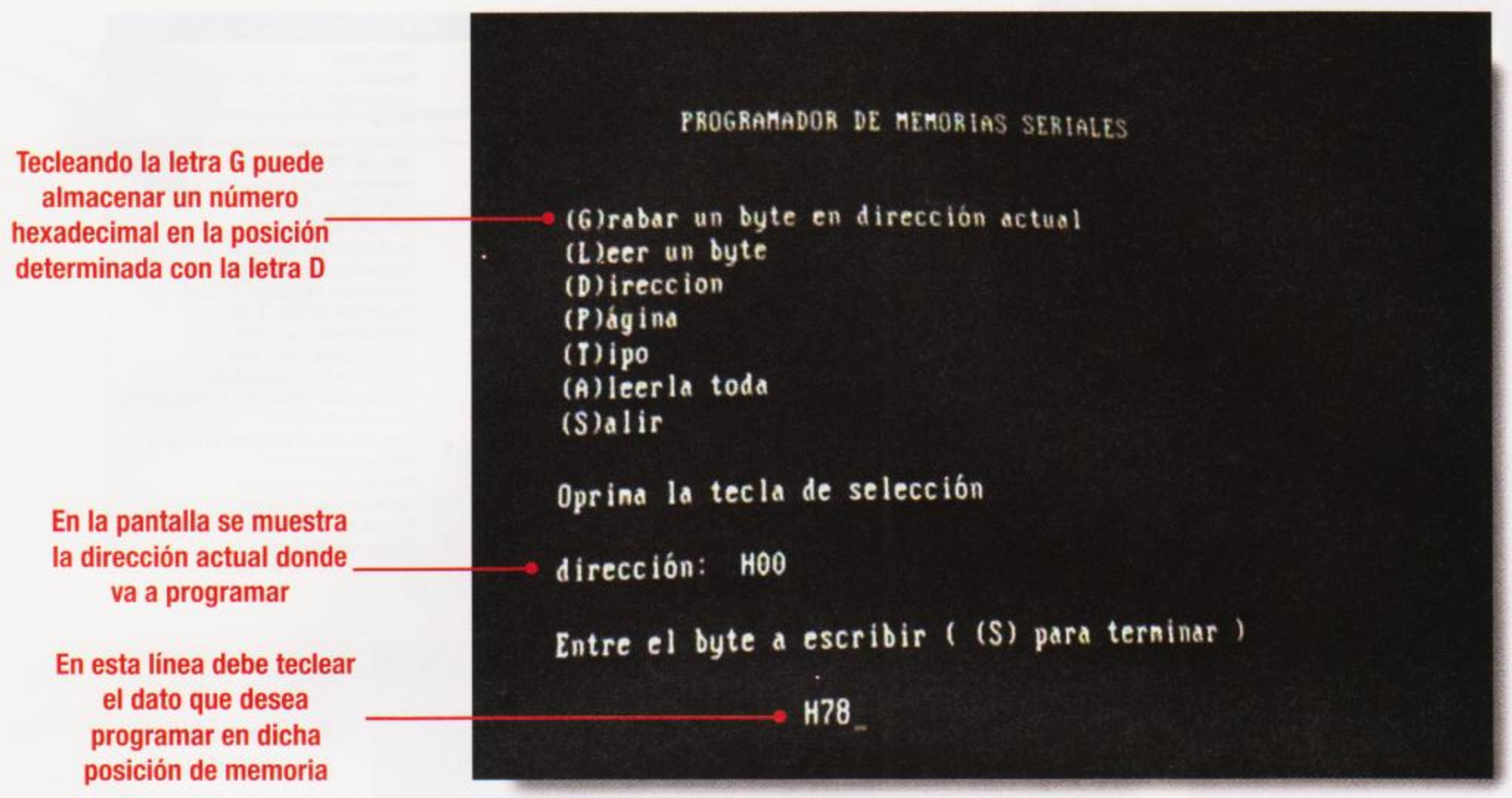


Figura 11.5e *Programación de la memoria*

moria escribir un nivel bajo indicando que la operación fue satisfactoria.

Lectura y escritura de los datos en forma serial

Se deben enviar los datos en formato serial acompañados de ciclos de reloj para su sincronización. La rutina básica fue denominada **enviabit**() que, como su nombre lo indica, permite el envío de un *bit* con sus respectivos pulsos de reloj hacia los pines de datos y reloj de las memorias.

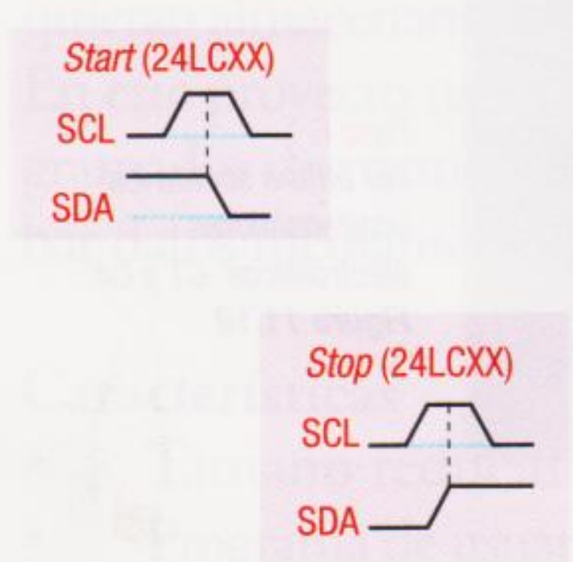


Figura 11.6 *Ciclos de inicio y finalización de operación de una memoria 24XX*

Así mismo, se ha elaborado la rutina **envia**() que recibe el dato con tamaño de un *byte* y lo organiza para enviarlo *bit* a *bit* por medio de la rutina anterior.

Al igual que con la escritura de los datos donde debe procesar un *byte* para enviar *bit* a *bit* hacia la memoria, para la lectura es necesario seguir el proceso contrario, es decir, recibir *bit* a *bit* y en la computadora conformar el *byte*.

El diagrama de tiempos para la lectura es el mismo para la escritura y aparece en la **figura 11.7**.

Lectura de datos en memoria 24XX

Para la secuencia de lectura de este tipo de memoria debe escribir una rutina en la cual debe hacer uso de la rutina anterior de lectura de un *byte* así como de las rutinas de *start*, *stop*, etc.

Todas las rutinas de lectura, reconocimiento y escritura sobre estas memorias se encuentran disponibles en nuestra página de la internet para que usted las aproveche y construya su propio programador.

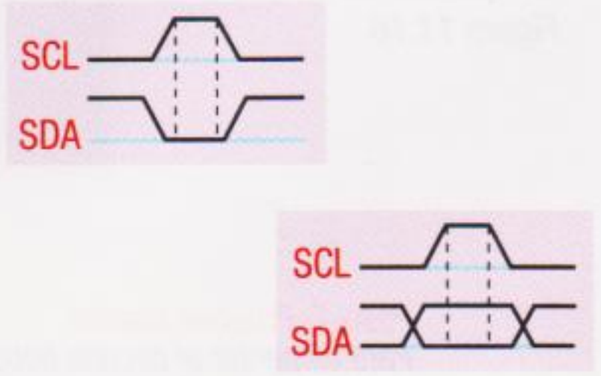


Figura 11.7 *Lectura y escritura serial de bits y bytes*

Ensamblaje del proyecto

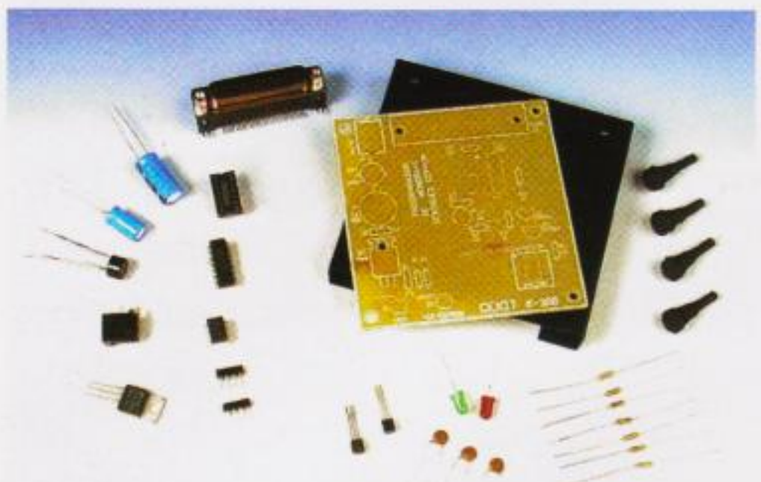


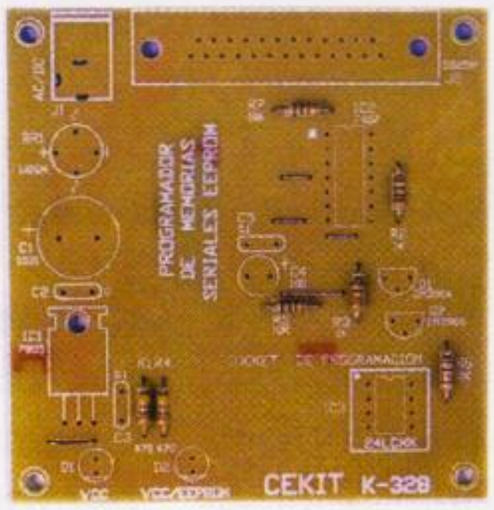
Figura 11.13 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el programador de memorias seriales EEPROM.*

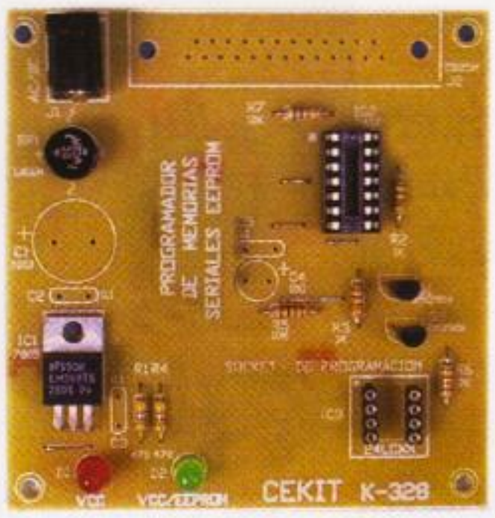
Este proyecto es una herramienta útil para programar las memorias seriales de la familia 24LCXX. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Regulador positivo fijo de +5V-1A, 7805
1	IC2	Circuito integrado 7407
1	IC3	Memoria serial EEPROM 24LC04
1	Q1	Transistor de propósito general NPN, 2N3904
1	Q2	Transistor de propósito general PNP, 2N3906
1	BR1	Puente rectificador circular de 1A, W06M
1	D1	Diodo LED rojo de 5mm
1	D2	Diodo LED verde de 5mm
1	C1	Condensador electrolítico de 1000uF - 25V
3	C2,C3 y C5	Condensadores cerámicos de 0.1uF - 50V
1	C4	Condensador electrolítico de 100uF - 25V
2	R1y R4	Resistencias de 470 - 1/4W
2	R2 y R3	Resistencias de 1k - 1/4W
3	R5 a R7	Resistencias de 10k - 1/4W
1	J1	Conector de alimentación AC/DC
1	J2	Conector DB25 macho para circuito impresos
1	K-328	Circuito impreso CEKIT
1		Base para circuito integrado de 14 pines
1		Chasis metálico K-328
4		Patas de caucho pequeñas
1		*Socket* en línea de 8 pines
4		Tornillos de 1/2"X1/4" con sus tuercas
4		Separadores plásticos

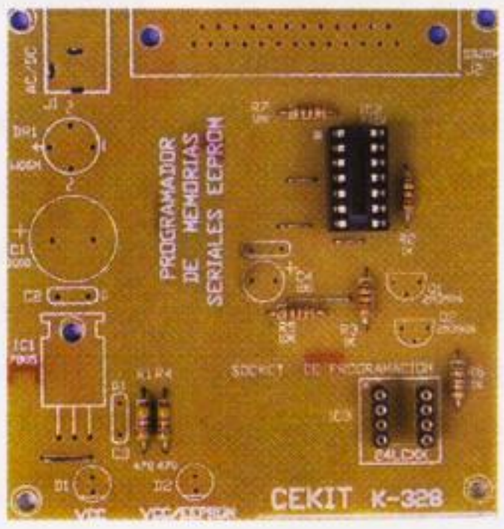
Paso 1
Con un alambre delgado, instale y suelde los puentes marcados en el circuito impreso, e instale y suelde también las resistencias.
Figura 11.14

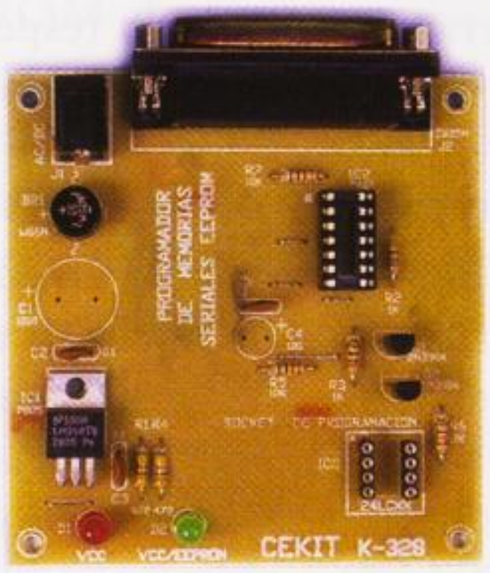




Paso 4
Instale y suelde el regulador de voltaje integrado IC1 (7805) y el conector de alimentación CA/CC.
Figura 11.17

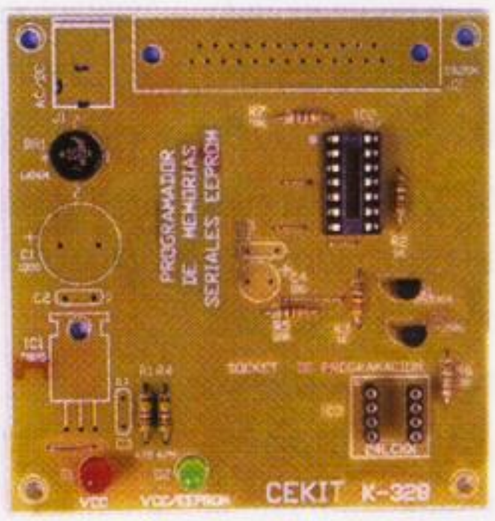
Paso 2
Luego instale y suelde la base para el circuito integrado IC2, los conectores (sockets) en línea para la memoria, y los dos diodos LED.
Figura 11.15

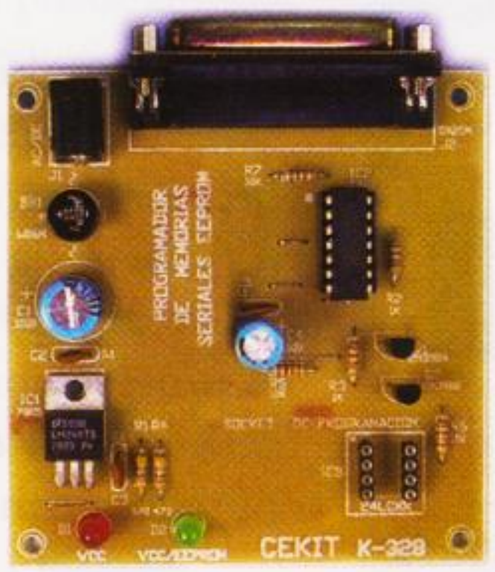




Paso 5
Instale y suelde el conector para el puerto paralelo DB-25 macho, en la posición indicada para J2, y los tres condensadores cerámicos de 0.1 µF .
Figura 11.18

Paso 3
Instale y suelde el puente rectificador circular BR1 y los transistores Q1 y Q2.
Figura 11.16





Paso 6
Por último suelde los condensadores electrolíticos C1 y C4.
Figura 11.19

Paso 7. Pruebas finales
Para alimentar el circuito debe usar una fuente CA/CC entre 10 y 20 V, una vez alimentado debe activarse el LED indicador de Vcc (ROJO). En el momento en que da la orden de programación desde la computadora, la alimentación a la memoria se activará, y se encenderá momentáneamente el LED indicador de programación (VERDE).

PROYECTO 12

GENERADOR DE MELODÍAS

En este proyecto construiremos un dispositivo generador de efectos musicales, que puede ser muy agradable para incorporarlo en nuestro hogar, sitio de trabajo, o simplemente como un obsequio.

Características

- Microcontrolador Motorola
- Tamaño reducido
- Emulación de notas musicales
- Salida por medio de parlante

Funcionamiento

La ciencia de los microcontroladores puede ser aplicada en infinidad de campos del conocimiento y la música no podía ser la excepción. Manipulando una señal de salida por un pin del microcontrolador y ajustando una escala de notas musicales adecuada, podemos generar una agradable melodía musical.

Muchos dispositivos modernos envían señales audibles a sus usuarios para indicarles alguna condición predeterminada o estado del sistema que se está controlando. Para ser agradables al oído humano, estos sonidos deben ajustarse a la escala musical. Muchos circuitos integrados en el mercado proporcionan estas capacidades de sonido, por ejemplo, un generador programable de sonido o un mensajero de voz como el ISD1016.

El circuito que se muestra en la **figura 12.1** no utiliza un generador de sonido dedicado, sólo genera sonidos mediante rutinas de programación. Como se puede apreciar su topología es muy sencilla, un microcontrolador con oscilador basado en cristal y un pin de salida de la señal en frecuencia que produce el sonido, aplicada sobre un parlante.

El circuito utiliza un microcontrolador MC68HC908JK3 de Motorola eliminando la necesidad de un circuito integrado de sonido. Tal vez muchos no estemos muy relacionados con el

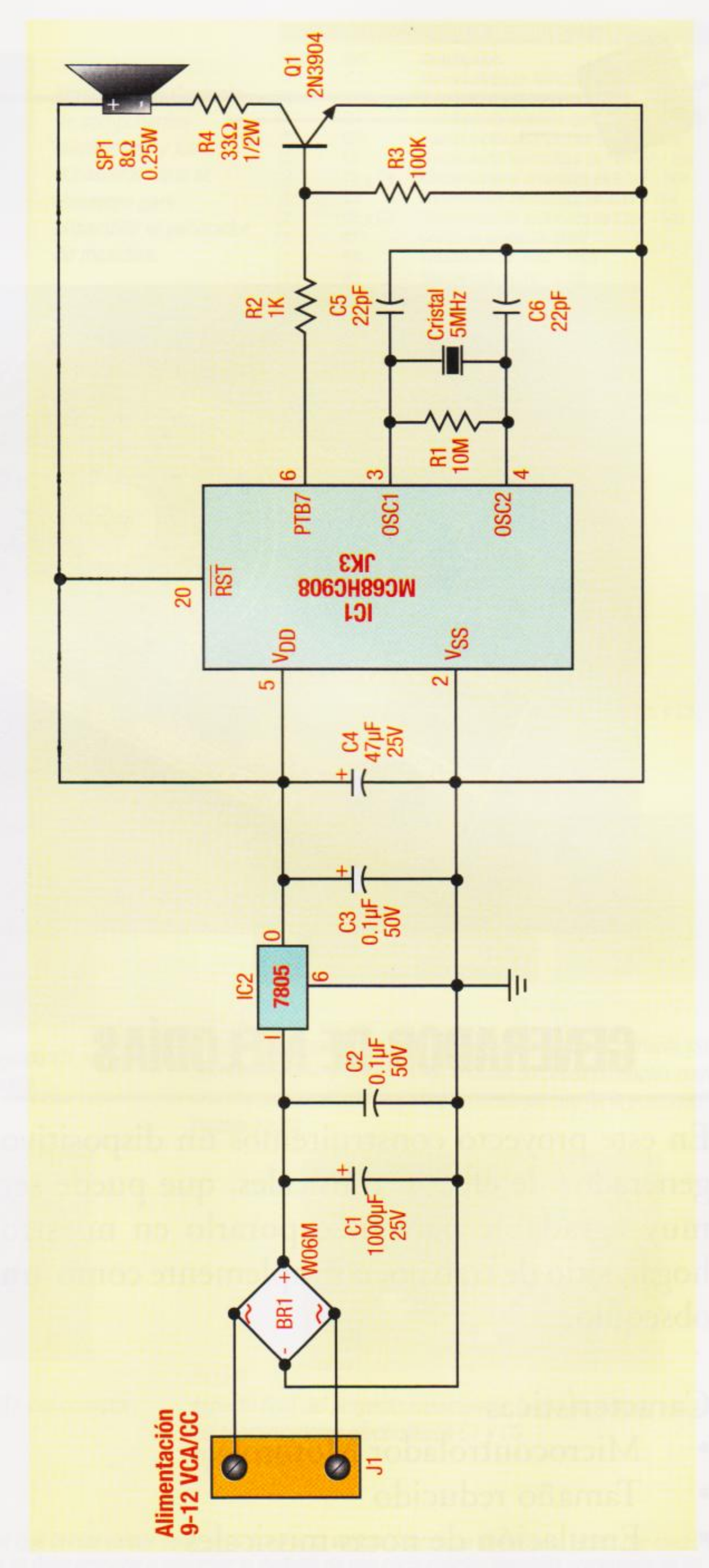
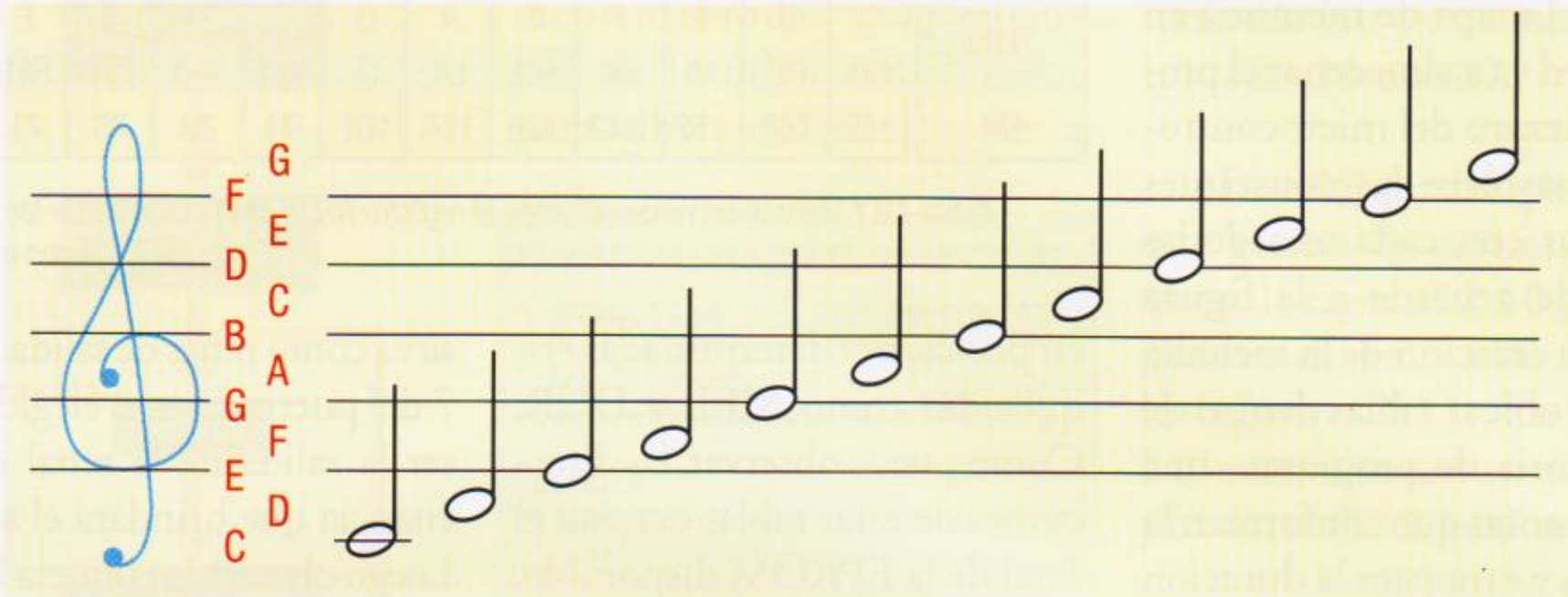


Figura 12.1 Diagrama esquemático del circuito



Nota	C	D	E	F	G	A	B	C	D	E	F	G
	DO	RE	MI	FA	SOL	LA	SI	DO1	RE1	MI1	FA1	SOL1
f, Hz	264	297	330	352	396	440	495	528	594	660	704	792
T, mSEC	3.78	3.36	3.04	2.84	2.51	2.3	2.0	1.9	1.68	1.5	1.4	1.26
1/2T, mSEC	1.89	1.68	1.52	1.42	1.26	1.14	1.01	0.94	0.84	0.76	0.71	0.63

Figura 12.2 *Tonos musicales con sus respectivas frecuencias y ubicación dentro del pentagrama*

mundo artístico, en especial el musical; para este caso tengamos en cuenta algunos conceptos.

Las melodías o pasajes musicales se conforman agrupando unidades denominadas notas o tonos musicales (DO, RE, MI, FA, SOL, LA, SI) los cuales podemos crear generando frecuencias determinadas desde el microcontrolador. En la **figura 12.2** podemos apreciar los tonos musicales con su respectiva ubicación dentro del pentagrama (dependiendo de la ubicación exacta dentro de estas cinco líneas, sabemos el tono correspondiente) y las frecuencias necesarias que se deben generar para originar satisfactoriamente el tono musical.

Existe otro parámetro importante que debe tener en cuenta, se trata de la duración que va a tener cada tono dentro de la melodía, para ello se han establecido unos símbolos que se colocan dentro del pentagrama y le dicen exactamente la duración que cada uno de ellos debe mantener. En la **figura 12.3** se muestran dichos símbolos que le indican la duración del tono. Ellos se denominan: redonda, blanca, negra, corchea, semicorchea (seguramente habrán más, por eso les pedimos disculpas a los expertos en la materia).

Descripción del programa

El primer paso en la programación, de un sistema de salida de sonido, es la elección del tipo de señal de sonido: sonidos individuales o una sección de una melodía. En este caso se han agrupado una serie de tonos y duraciones en tablas ubicadas en memoria, las cuales se van enviando de forma secuencial por un pin del microcontrolador hacia el dispositivo de salida (parlante).

La primera parte del programa, **figura 12.4**, además de

Nota					
Duración	Redonda	Blanca 1/2	Negra 1/4	Corchea 1/8	Semicorchea 1/16
DR	16	8	4	2	1

Figura 12.3 *Duración de las distintas notas musicales*

definir el mapa de memoria en que usted va a almacenar el programa dentro del microcontrolador, establece las constantes asociadas con cada una de las notas, de acuerdo a la **figura 12.2**. La creación de la melodía implica ubicar tablas dentro de la memoria de programa, una para las notas que conforman la melodía y otra para la duración que debe tener cada una de ellas. Estas direcciones se establecen en posiciones determinadas etiquetadas como MEL y DUR. Como puede observar, las direcciones de estas tablas ocupan el final de la EPROM disponible.

Nota	C	D	E	F	G	A	B	C	D	E	F	G
	DO	RE	MI	FA	SOL	LA	SI	DO1	RE1	MI1	FA1	SOL1
NR	189	168	152	142	126	114	101	94	84	76	71	63

Tabla 12.1 *Valores correspondientes al registro NR para la creación de los distintos tonos musicales*

El segundo bloque del programa es el que está en la **figura 12.5**, en este código se destaca la definición de las variables NR y DR. NR corresponde al registro donde está el valor necesario para crear cada uno de los tonos musicales de acuerdo a la **tabla 12.1**, y DR es el registro donde se almacena la escala de tiempo necesaria para el buen funcionamiento de cada símbolo musical, de acuerdo a lo dispuesto en la **figura 12.3**.

El tercer bloque del programa es el lazo principal de todo el proyecto, **figura 12.6**. Inicialmente se configura el funcionamiento del puerto D para que le sirva como pines de salida. El pin 7 del puerto D es el elegido para ser la salida de la señal en frecuencia que brindará el sonido. Luego observe la etiqueta PRINCIPAL y unas cuantas líneas de código; este pequeño conjunto conforma el bloque principal del programa, en el cual, simplemente, se invocan dos rutinas, una de retardo y una de generación de la melodía, las cuales se repiten continuamente, como se muestra en la **figura 12.7**.

```
; CURSO PRÁCTICO SOBRE
; MICROCONTROLADORES
; CEKIT S.A.
; GENERADOR DE MELODÍAS

$Include 'jl3regs.inc'
RAMStart      EQU  $0080
RomStart      EQU  $EC00
RomFin        EQU  $F000
RESET_VEC     EQU  $FFFE
;Reset Vector
; CONSTANTES ASIGNADAS A CADA
; TONO MUSICAL
DO            EQU 189T
RE            EQU 168T
MI            EQU 152T
FA            EQU 142T
SOL           EQU 126T
N             EQU 26T ; NÚMERO
; DE NOTAS DE LA MELODÍA
ND            EQU 100T; NÚMERO
;DE REPETICIONES DE LA MELODÍA
SND           EQU 7
DUR           EQU RomFin-N
;TABLA DE DURACIÓN DE LOS TONOS
MEL           EQU RomFin-N-N ;
; TABLA DE LOS TONOS DE LAS
; NOTAS. AMBAS TABLAS SE
; UBICAN AL FINAL DE LA
; MEMORIA DE PROGRAMA.
```

Figura 12.4 *Definición de las constantes de cada tono musical*

```
org RamStart
NR        RMB 1 ;REGISTRO DEL
            ;TONO DE UNA NOTA
DR        RMB 1 ; REGISTRO DE
            ;DURACIÓN DEL TONO
NDR       RMB 1 ; REGISTRO DE
          ; DURACIÓN DE LA NOTA
MEMX      RMB 1
MEM1      RMB 1
    org RomStart
```

Figura 12.5 *Definición de las variables dentro de la RAM*

```
*****************************
;  - DESDE ESTE PUNTO SE
;INICA LA EJECUCION DEL
;CODIGO DESPUES DE UN RESET.
*
*****************************
INICIO:
       RSP ;INCIO LA PILA
       CLRA
       CLRX
       MOV #$FF,DDRD ; CONFI-
      ;GURO EL PUERTO D COMO
      ;SALIDAS
       CLR PORTD ; LO BORRO

PRINCIPAL
       JSR RETARDO
       JSR MELODIA ; EMPIEZA
                   ; LA MÚSICA.
       JSR RETARDO; ESTE ES
       JSR RETARDO; EL LAZO
       BRA PRINCIPAL; PRINCIP.
```

Figura 12.6 *Cuerpo principal del programa*

Figura 12.7 *Diagrama de flujo del lazo principal del programa*

La subrutina MELODIA es la que se encarga dentro del programa de generar el pasaje musical que busca, su código es el que se muestra en la **figura 12.8** y su diagrama de flujo el de la **figura 12.9**. Este segmento inicializa un apuntador (x) hacia cada posición de la tabla, desde donde se toman los valores de tono y duración y se almacenan en los registros NR y DR; se invoca una rutina para generar la nota y se ejecuta hasta que se haya cumplido el tiempo establecido.

Finalmente el apuntador se incrementa para acceder a la siguiente posición de memoria, desde donde se toman los segundos valores para repetir la generación de la nota. El proceso se repite hasta que el apuntador haya recorrido todas las posiciones de la tabla establecidas para albergar las notas de la melodía (en este caso se han tomado 26 notas).

```
****************************
* SUBRUTINA DE GENERACIÓN DE*
* LA MELODÍA                 *
****************************
MELODÍA:

        CLRX    ; 0 -> X
M1      LDA     MEL,x ; CARGO
        ;EL TONO DE LA NOTA
        STA     NR    ; EN EL
        ;REGISTRO NR
        LDA     DUR,x ; CARGO
        ;LA DURACIÓN DEL TONO
        STA     DR    ; DE LA
      ;NOTA EN EL REGISTRO DR
        LDA     #ND   ; CARGO
    ;EL NÚMERO DE REPETICIONES
        STA     NDR   ; DE LA
     ; NOTA EN EL REGISTRO NDR
        STX     MEMX  ;
        ; GUARDO X EN MEMORIA
M2      JSR     NOTA  ;
        ; GENERO LA NOTA
        DEC     DR
        BNE     M2
        BCLR    SND,PORTD ;
     ; HAGO UNA PAUSA DE 10 ms
        LDX     #100T ; ANTES
        ; DE CADA NOTA
        JSR     RETAR
        LDX     MEMX ;
; RECARGO X DESDE LA MEMORIA
        CPX     #N-1 ; HAY
        ; MÁS NOTAS?  n-1
        BLO     M3
        RTS     ; FIN DE LA
        ;SUBRUTINA DE MELODÍA
M3      INCX    ; VOY A LA
        ;SIGUIENTE NOTA
        BRA     M1
```

Figura 12.8 *Subrutina que genera la melodía*

El código del programa que genera las notas se aprecia en la **figura 12.10** y su respectivo diagrama de flujo en la **figura 12.11**. En ella se hace una conmutación del pin de salida que se repite

Figura 12.9 *Diagrama de flujo de la subrutina MELODIA*

mientras el registro que indica la duración se decrementa y llega a cero.

En la **figura 12.12** se observa una sencilla rutina de retardo por *software* establecida para ser llamada desde otras partes del programa.

Ella se construye de manera similar a las que hemos discutido en proyectos anteriores: una serie de ciclos que se ejecutan continuamente mientras el contenido de unos registros se van decrementando hasta llegar a cero.

```
*****************************
*GENERACIÓN DE LA FRECUENCIA
* O TONOS MUSICALES
*****************************
NOTA    BSET    SND,PORTD ;
        ;EMPIEZO UNA NOTA
        LDX     NR
        JSR     RETAR
        BCLR    SND,PORTD
        LDX     NR
        JSR     RETAR
        LDA     NDR
        BEQ     NO
        DEC     NDR
        BRA     NOTA
NO      RTS     ; FINAL DE LA
                ; NOTA

RETAR   LDA     #2 ; RETARDO
                   ;DE 0.01 ms
REPO    DECA       ;SIN UTILI-
        BNE REPO   ;SACIÓN DEL
        ; TEMPORIZADOR TIM.
        DECX ; SÓLO CICLOS
        BNE RETAR ; DE SOFT-
        RTS        ; WARE
```

Figura 12.10 *Subrutina que genera las notas musicales*

Figura 12.11 *Diagrama de flujo de la subrutina generadora de las notas*

```
*****************************
* RETARDO- SUBRUTINA DE
RETARDO POR SOFTWARE *
*****************************
RETARDO:
        LDA     #10T
        STA     MEM1
ET1     JSR     RETITO
        DEC     MEM1
        BNE     ET1
        RTS
RETITO:
        LDA     #128T ;
        ;RETARDO DE 0.1 s 128
ET2     CLRX
ET3     DECX
        BNE     ET3
        DECA
        BNE     ET2
        RTS
```

Figura 12.12 *Retardo por software*

En la **figura 12.13** observe el último bloque del programa, en él se especifican las tablas localizadas en memoria en las cuales se ubican los distintos tonos y duraciones que componen la melodía que se pretende desplegar.

```
*****************************
* UBICACIÓN DE TABLAS EN
* MEMORIA                 *
*  AQUÍ SE UBICA LA COMPOSI-*
* CIÓN MUSICAL
*****************************
ORG  MEL
FCB MI,MI,MI,MI,MI,MI,MI,
FCB SOL,DO, RE, MI,
FCB FA,FA,FA,FA,FA,MI,MI,
FCB MI,MI,MI,RE,
FCB  RE,MI,RE,SOL
ORG  DUR
FCB  2,2,4,2,2,4,2,2,2,2,8,
FCB  2,2,2,2,2,2,2,1,1,2,2,
FCB  2,2,4,4
Org RESET_VEC
FDB INICIO    ;al ordenarse reset
              ;salta a Start
END
```

Figura 12.13 *Estructura de notas que conforman el pasaje musical*

Para crear sus propias melodías debe modificar la variable N de la figura 12.4, de acuerdo al número de notas que vaya a colocar. Luego, en esta parte del programa debe especificar cuales notas (tonos) componen su canción en la tabla MEL y los tipos de notas (duración) correspondientes a cada uno de esos tonos, los cuales se escriben dentro de la tabla DUR. El número de elementos de cada tabla debe coincidir con el valor de la variable N.

PARA SABER MÁS...

Comandos del simulador ICS08JL para microcontroladores Motorola

El programa ICS08JL junto con el editor WINIDE (desarrollado por P&E Microcoputers) conforman un entorno que simplifica enormemente el proceso de escritura y depuración del código fuente. Con este paquete los diseñadores están en capacidad de crear códigos fuente, ensamblar el código, depurarlo y programar los microcontroladores 68HC908JL3/JK3/JK1.

Dentro del entorno del simulador se destaca una línea de comandos donde el usuario puede entrar mediante el teclado toda una serie de órdenes sobre el sistema y sobre el programa, **figura 12.14**. Observemos algunos de ellos.

LOAD

El comando LOAD carga un archivo S19 dentro del entorno del programa para operar sobre él. Recuerde que un archivo S19 es el resultante de ensamblar un código fuente. Si se teclea la palabra LOAD se desplegará una ventana con todos los archivos S19 del directorio actual. La sintaxis normal es:

LOAD [nombre_del_archivo].

RESET

Este comando simula un estado de *reset* en el microcontrolador y pone en el contador de programa los contenidos del vector de *reset*. Este comando no inicializa de nuevo la ejecución del programa; si ese es su deseo debe teclear la instrucción **RESETGO**.

GO

Inicia la ejecución del código a partir de la actual dirección indicada en el contador de programa. Opcionalmente se pueden establecer las direcciones de inicio y fin que se desean ejecutar. Su sintaxis es:

GO [<dirección_inicio> [<dirección_final>]]

PC

El comando PC asigna un valor específico al contador de programa. Como el PC indica la dirección de la siguiente instrucción que se va a ejecutar, al asignar un nuevo valor de PC se modifica el flujo de la ejecución del código. Sintaxis:

PC <dirección>

MD

Este comando se utiliza para visualizar en la ventana de memoria del entorno los contenidos de ciertas posiciones de memoria. El número de *bytes* visualizados depende del tamaño de la ventana. Su sintaxis es:

MD <dirección>

STACK

Al teclear este comando se despliega una nueva ventana donde se puede visualizar la pila del HC08. En esta ventana se muestra el valor del puntero a la pila (SP: *stack pointer*) y el dato almacenado en la pila. Sintaxis:

STACK

STEP

Este comando permite correr paso a paso las instrucciones de un programa cargado en la memoria del ICS08JL, iniciando en la dirección especificada en ese momento por el PC. Todas las ventanas se actualizan cada vez que este comando se ejecuta. Este comando puede teclearse solo o junto con un parámetro (n) que indica el número de instrucciones que debe ejecutar. Sintaxis:

STEP [<n>]

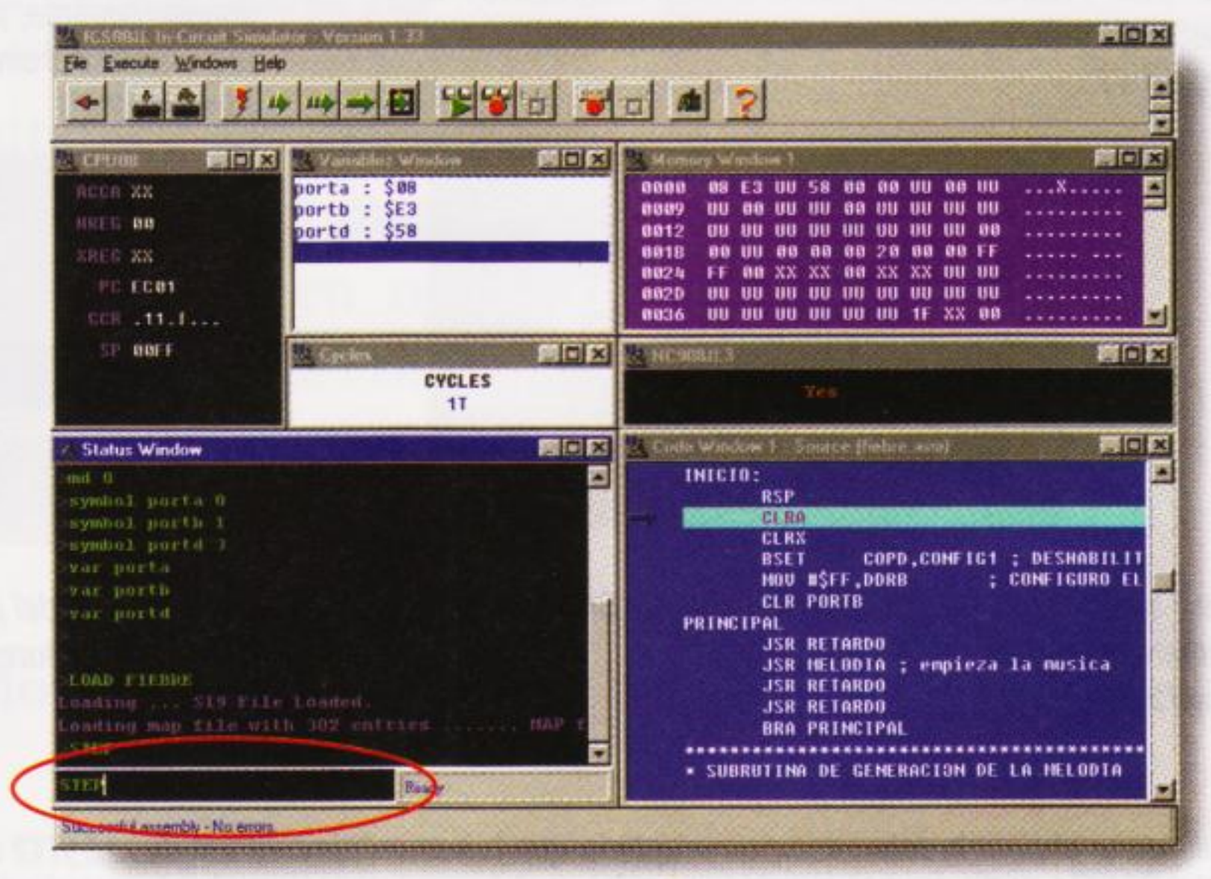


Figura 12.14 Entorno del simulador ICS08JL

Ensamblaje del proyecto

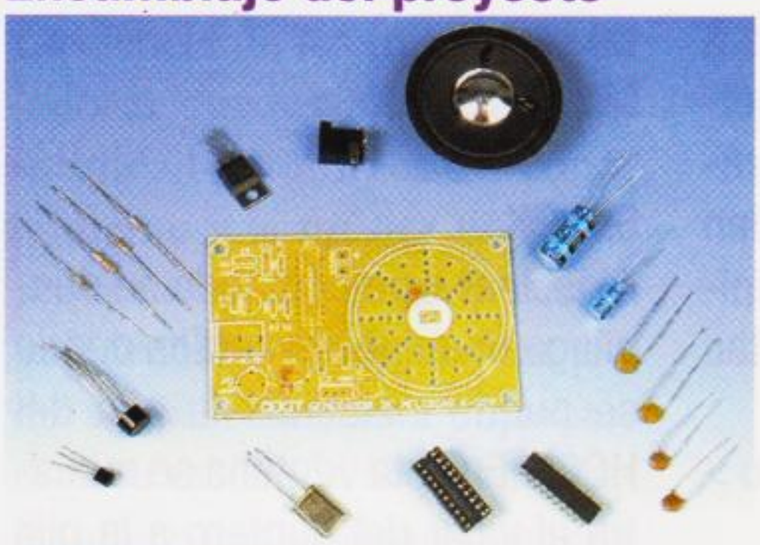
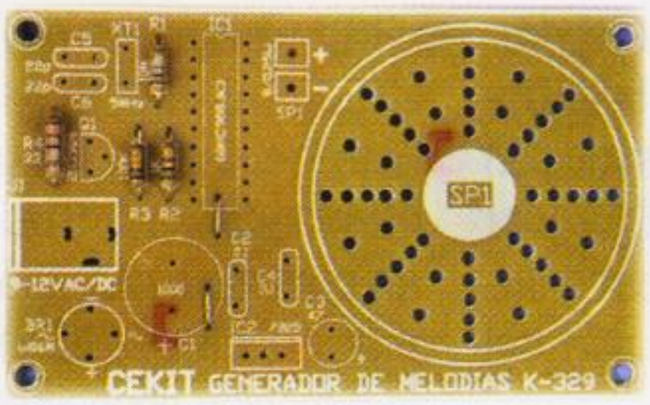


Figura 12.15 Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el generador de melodías.

LISTA DE MATERIALES		
Cant.	**Ref.**	**Descripción**
1	IC1	Microcontrolador 68HC908JK3
1	IC2	Regulador positivo fijo de +5V-1A, 7805
1	Q1	Transistor de propósito general NPN, 2N3904
1	BR1	Puente rectificador circular de 1A, W06M
1	C1	Condensador electrolítico de 1000uF - 25V
2	C2 y C4	Condensadores cerámicos de 0.1uF - 50V
1	C3	Condensador electrolítico de 47uF - 25V
2	C5 y C6	Condensadores cerámicos de 22pF - 50V
1	XT1	Cristal de quarzo de 5MHz
1	R1	Resistencia de 10M - 1/4W
1	R2	Resistencia de 1k - 1/4W
1	R3	Resistencia de 100k - 1/4W
1	R4	Resistencia de 33 - 1/2W
1	SP1	Parlante pequeño de 8Ohm-0.25W
1	K-329	Circuito impreso CEKIT
1		Conector de alimentación AC/DC
1		Base para circuito integrado de 20 pines

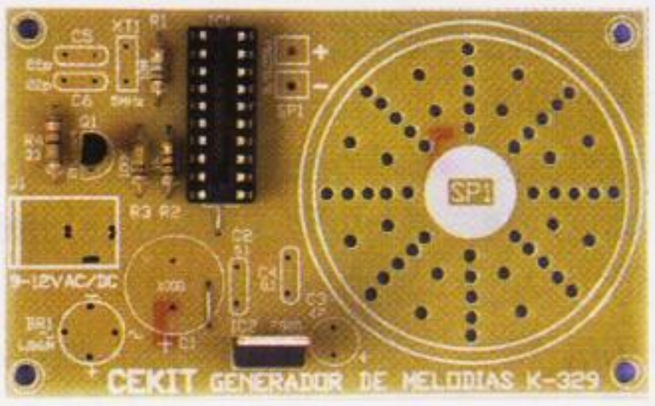
Con este proyecto usted podrá divertirse escuchando y programando sus propias melodías, a través del lenguaje de programación de los microcontroladores Motorola. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:



Paso 1

Instale y suelde los puentes de alambre y las resistencia R1, R2, R3 y R4.

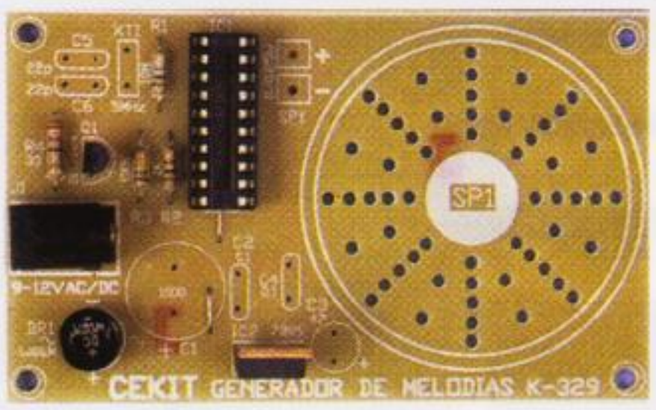
Figura 12.16



Paso 2

Luego instale y suelde la base para el circuito integrado IC1, el transistor Q1 y el regulador de voltaje integrado IC2.

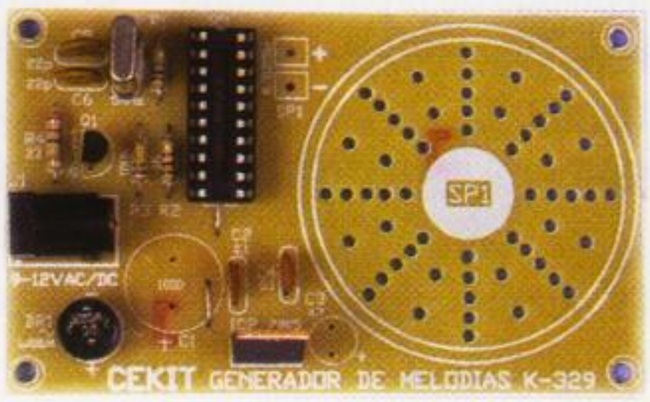
Figura 12.17



Paso 3

Instale y suelde el puente rectificador circular BR1 y el conector de alimentación CA/CC (J1).

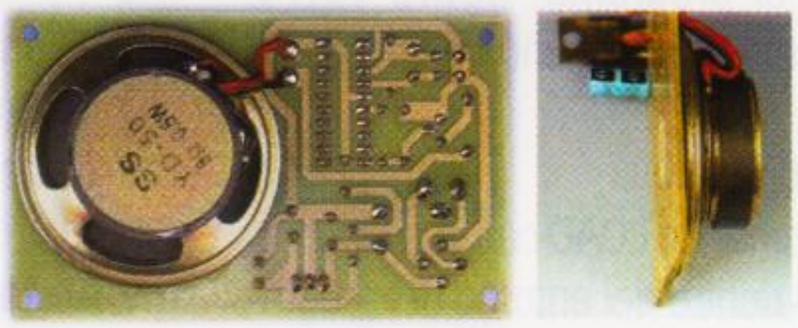
Figura 12.18



Paso 4

Instale y suelde los condensadores cerámicos C2, C4, C5 y C6, y el cristal de cuarzo XT1.

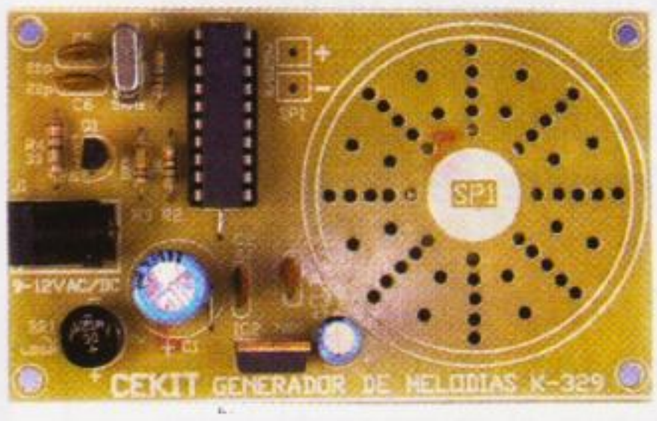
Figura 12.19



Paso 5

Asegure por la parte posterior del circuito impreso el parlante (SP1) usando pegante, y luego suelde el cable de conexión duplex entre el parlante y el circuito impreso, teniendo en cuenta la polaridad.

Figura 12.20



Paso 6

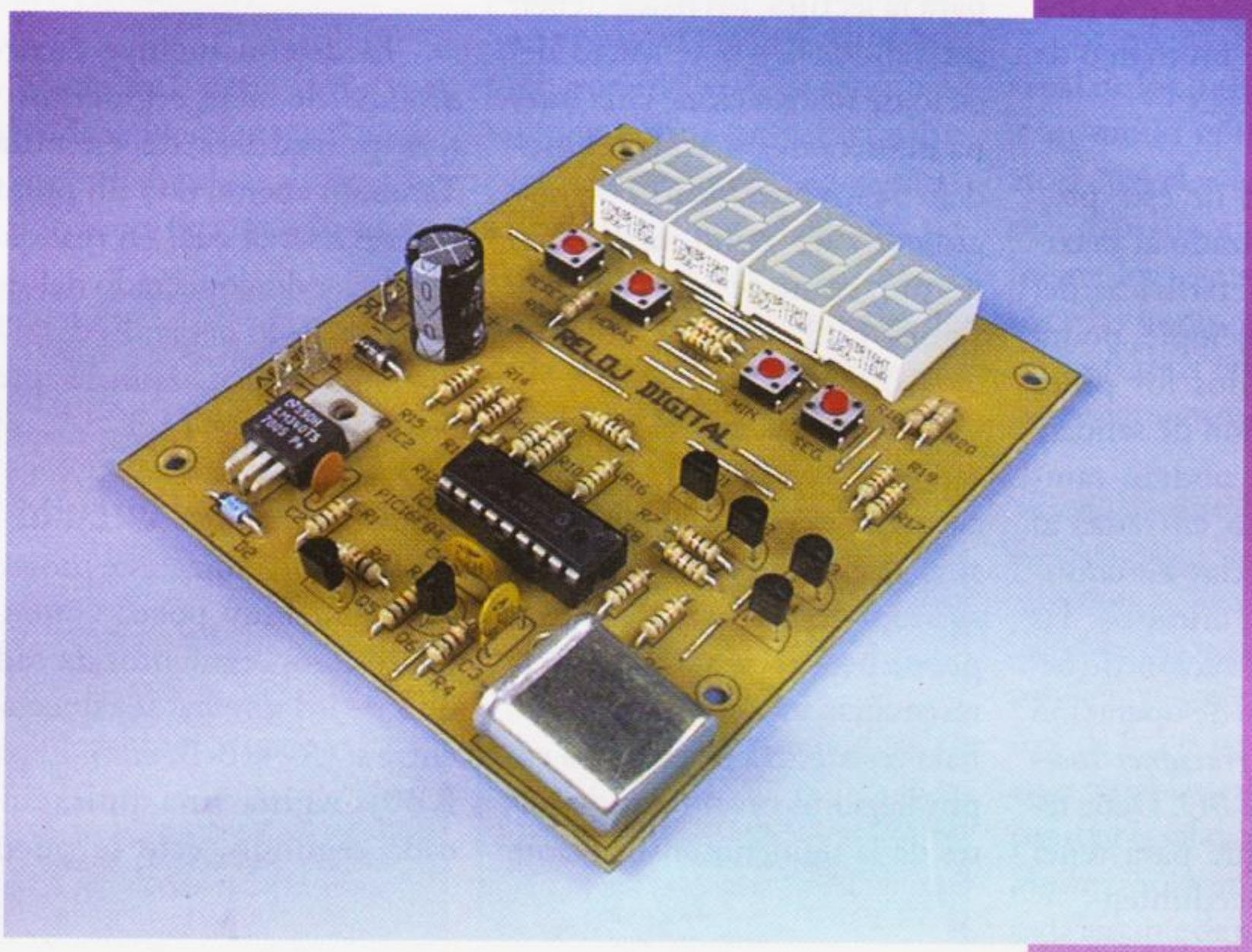
Aspecto final del proyecto ensamblado, luego de instalar y soldar los condensadores electrolíticos C1 y C3.

Figura 12.21

Paso 7. Pruebas finales

Para alimentar el circuito debe usarse un adaptador que brinde voltaje continuo entre 8 y 12 voltios. Cuando se alimenta la tarjeta por medio del conector CA/CC a partir de unos cuantos segundos se debe empezar a escuchar la melodía de una corta canción navideña compuesta de 26 notas musicales. Si no es así, revise que el microcontrolador tenga los voltajes apropiados de alimentación y que se encuentre programado adecuadamente.

PROYECTO 13



RELOJ DIGITAL CON PIC

En esta ocasión usted construirá un reloj digital que puede adaptarse para gran cantidad de tareas de control e instrumentación. También puede ser muy útil para instalarlo en su automóvil, hogar o sitio de trabajo.

Características

- Reloj de tiempo horario real
- Visualización de horas, minutos y segundos
- Error aproximado de 1.2 segundos al año
- Visualización en cuatros displays de siete segmentos
- Pulsadores para el ajuste de la hora actual
- Batería de respaldo para cortes de energía
- Bajo costo

Esta aplicación particular de los microcontroladores PIC es una buena solución para todos los desarrolladores de proyectos, al encontrar el problema de diseñar sistemas que funcionen dentro de un marco de tiempo horario real.

En el diseño de este proyecto se han utilizado al máximo algunas características del PIC16F84. Por ejemplo, podrá observar que todos los pines cumplen la función de salida y algunos de ellos podrán también servir como entradas al poder leer las teclas de configuración de los minutos, las horas y la visualización de segundos. Este tipo de operación se denomina *operaciones sucesivas en un puerto I/O*. Debe tener en cuenta que para tener éxito en este procedimiento de ejecutar una operación de escritura, seguida de una instrucción de lectura en el mismo puerto, la secuencia de las instrucciones debe ser tal que permita estabilizar el voltaje de los pines (dependiendo de la carga), antes de que la siguiente instrucción inicie la siguiente lectura hacia la CPU. Dicho de otra manera, puede ocurrir que la configuración previa del puerto sea la que la CPU entienda en lugar de la nueva configuración. Si se presenta alguna duda, es mejor asegurarse con una instrucción NOP. La **figura 13.1** muestra el diagrama de tiempos que ilustra las operaciones sucesivas en un puerto I/O. La escritura actual en un puerto I/O ocurre al final de un ciclo de instrucción, mientras que para la lectura, los datos deben ser válidos desde el inicio del ciclo de instrucción. Una buena manera de entender el modelo de los puertos I/O es imaginando un condensador en un pin, como se muestra en la **figura 13.2**.

Debido a que el condensador necesita un tiempo para cargarse, el pin también necesita el mismo tiempo. Para superar este inconveniente puede pensar en adicionar una resistencia en serie al pin I/O. Esta resistencia permitirá a ese pin llegar al nivel deseado antes de la siguiente instrucción. El uso de instrucciones NOP entre las subsecuentes instrucciones de paso de lectura a escritura del puerto es una solución más económica.

El diseño incluye cuatro *displays* de siete segmentos y tres pulsadores de entrada. También cuenta con un pulsador de *reset* el cual en realidad puede ser desconectado del sistema, si así lo desea. El diagrama esquemático lo puede apreciar en la **figura 13.3**.

Descripcion del circuito

Como podrá notar, este proyecto utiliza muy pocos componentes. Los segmentos de cada uno de los *displays* se conectan entre sí (A-A, B-B, etc.). El pin RB0 origina una pulsación cada segundo, éste se puede

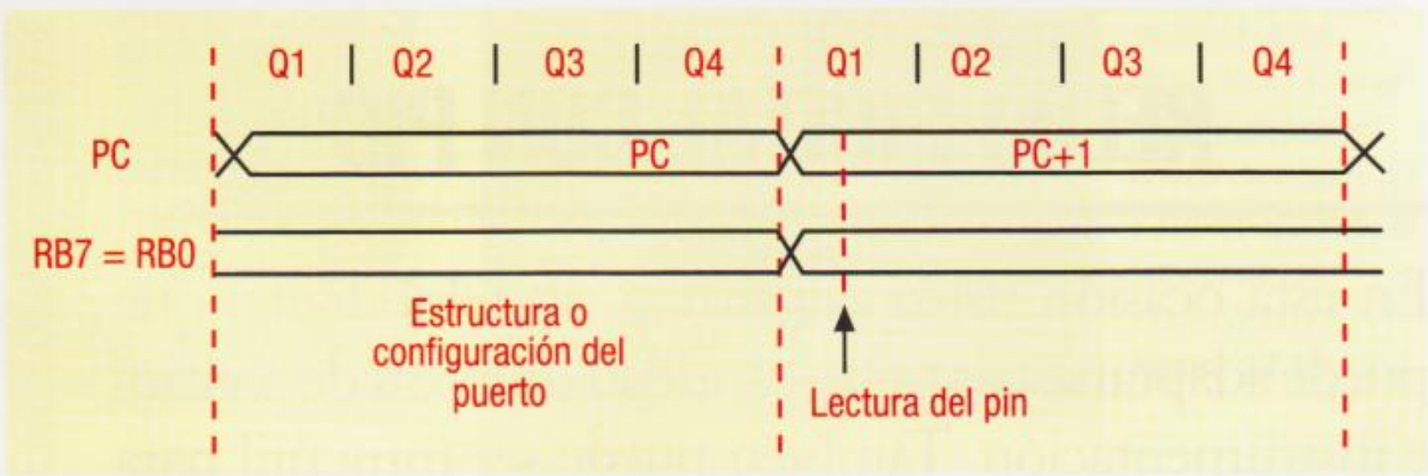


Figura 13.1 *Operaciones sucesivas en un puerto I/O*

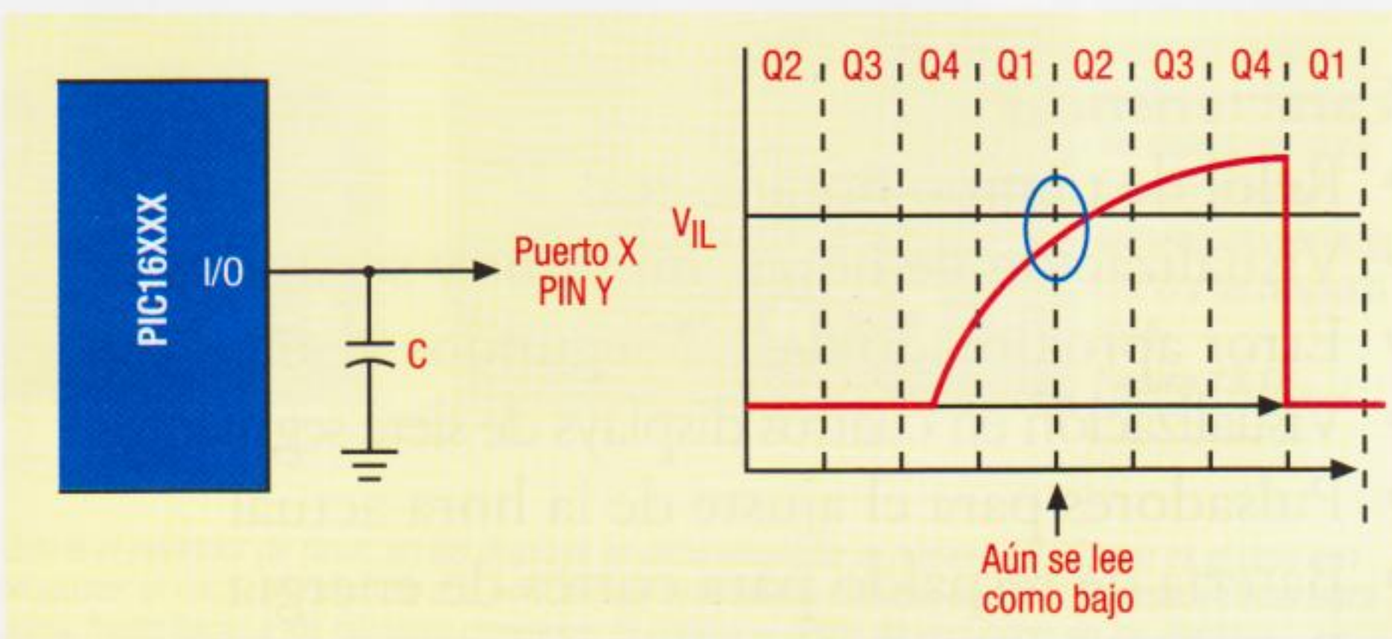


Figura 13.2 *Comportamiento del puerto*

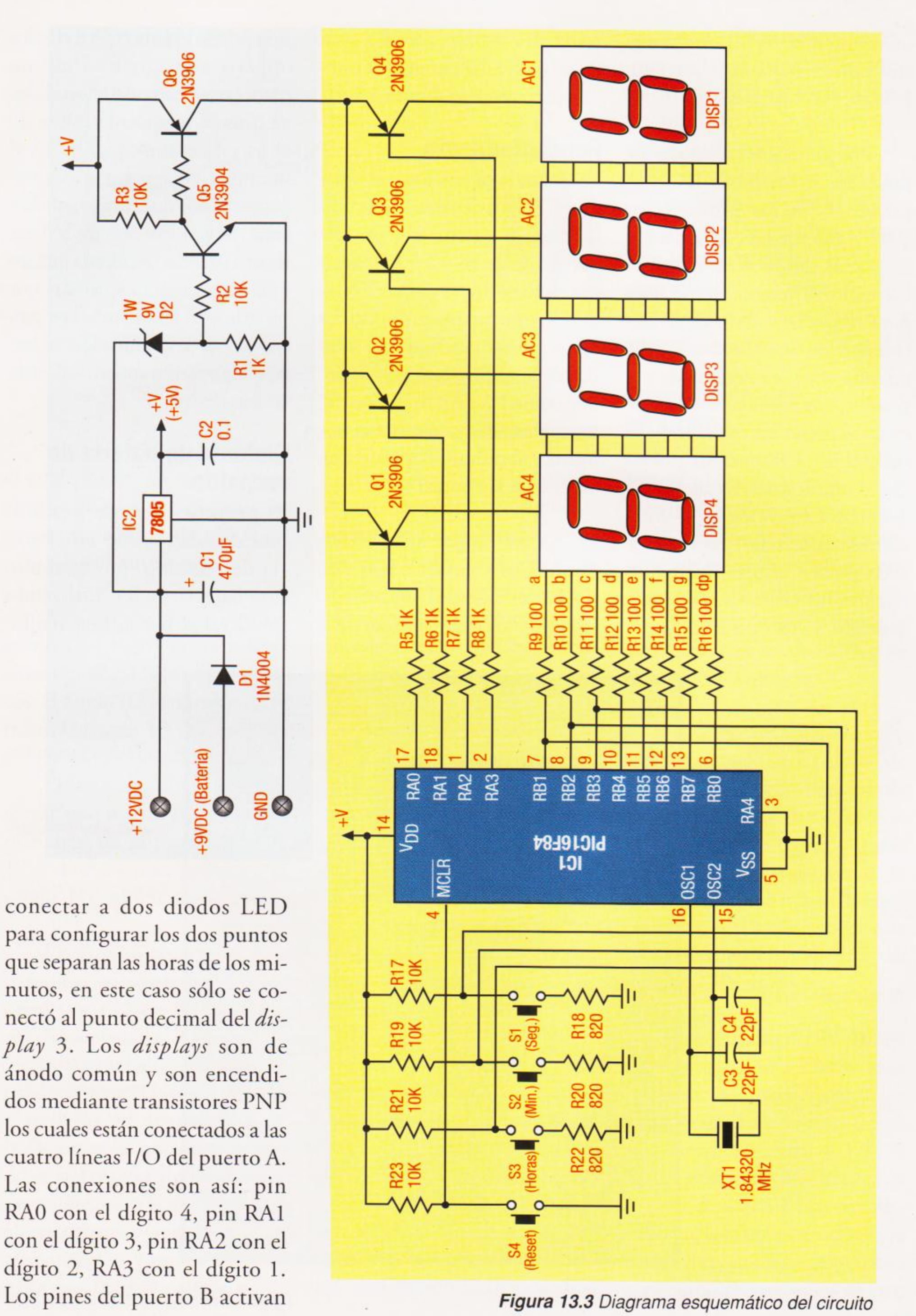


Figura 13.3 *Diagrama esquemático del circuito*

conectar a dos diodos LED para configurar los dos puntos que separan las horas de los minutos, en este caso sólo se conectó al punto decimal del *display* 3. Los *displays* son de ánodo común y son encendidos mediante transistores PNP los cuales están conectados a las cuatro líneas I/O del puerto A. Las conexiones son así: pin RA0 con el dígito 4, pin RA1 con el dígito 3, pin RA2 con el dígito 2, RA3 con el dígito 1. Los pines del puerto B activan

los segmentos de los *displays*, y sus conexiones son RB1-a, RB2-b, RB3-c...RB7-g.

Los pulsadores se conectan también a los pines I/O del puerto B. A los pines RB1, RB2 y RB3 se les conectan resistencias de 10 KΩ. Este valor es lo suficientemente alto para impedir el paso de la corriente de los diodos LED cuando ellos se encuentren activados. Las entradas son detectadas al poner los pines en bajo mediante el respectivo pulsador y una resistencia de 820Ω a tierra. Este valor es lo suficientemente bajo para poner cada pin en alto rápidamente cuando las salidas se han desactivado, y crear un 90% de entrada alta debido a V. La guía de montaje del circuito la puede apreciar en la **figura 13.4.**

Funcionamiento

Al iniciar la operación o al oprimir el botón de *reset*, el circuito iniciará la cuenta del tiempo partiendo de las 12:00. Si oprime el S1 podrá visualizar los segundos que están corriendo. La hora puede configurarla mediante los pulsadores referenciados como S2 y S3. Al pulsar S2 se origina un avance en los minutos y al pulsar S3 se causa un avance en las horas.

Debido a que todos y cada uno de los segmentos han sido unidos en todos los *displays*, sólo uno de ellos debe encenderse al tiempo, de lo contrario todos los *displays* mostrarían el mismo dato. Los *displays* son encendidos de derecha a izquierda, haciendo el barrido o multiplexaje en el instante en que tenga listo el respectivo valor que irá asignando a cada *display*. Ese barrido se hace de una manera tan rápida que sus ojos no lo notarán y pensará que no existe tal fenómeno. Los pulsadores de configuración se leerán una vez terminado cada ciclo de los *displays*.

Sistema de batería de respaldo

El proyecto tiene la posibilidad de adicionarle una batería de respaldo, por ejemplo, una batería de 9V utilizando un sistema bastante sencillo,

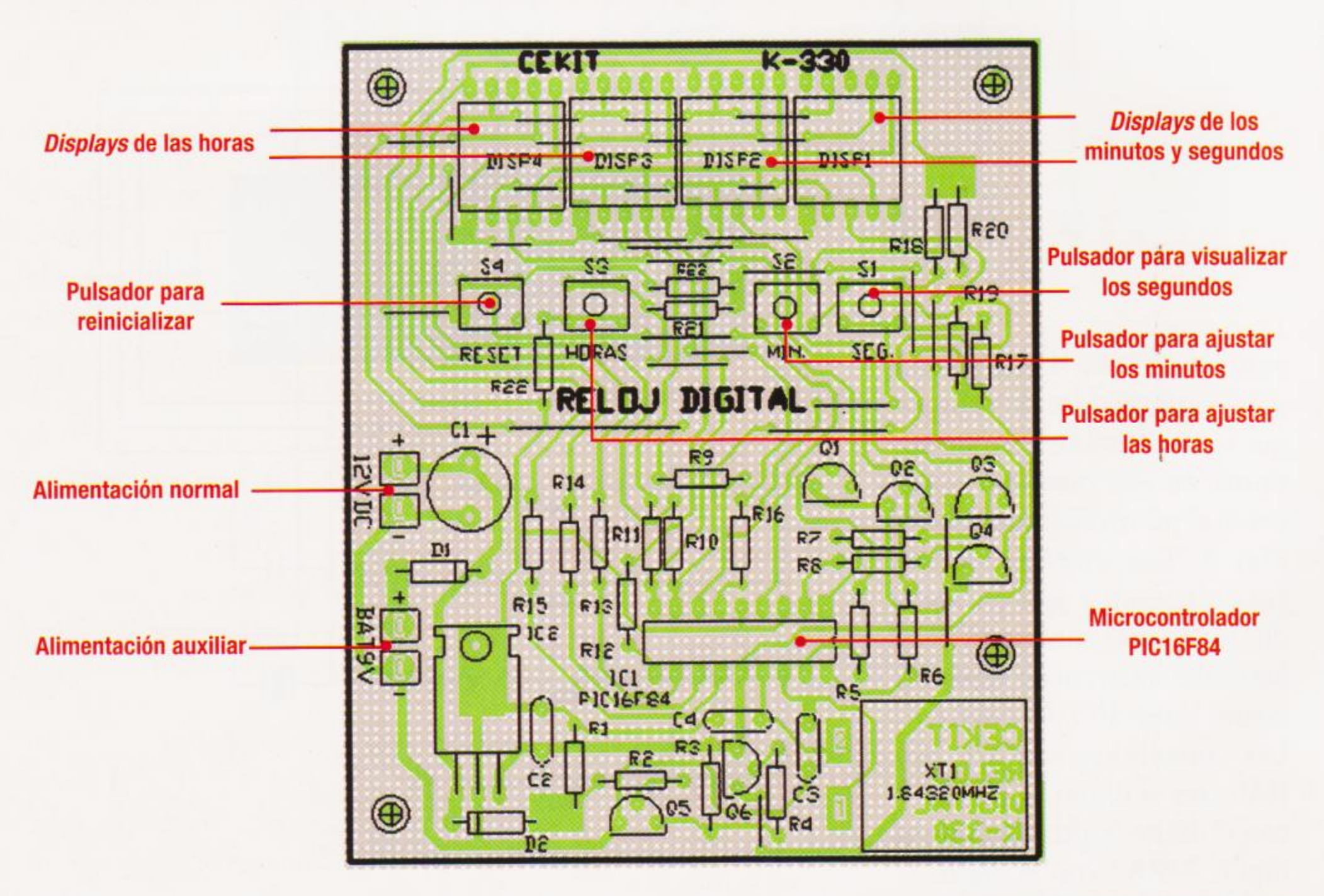


Figura 13.4 Guía de montaje del reloj digital

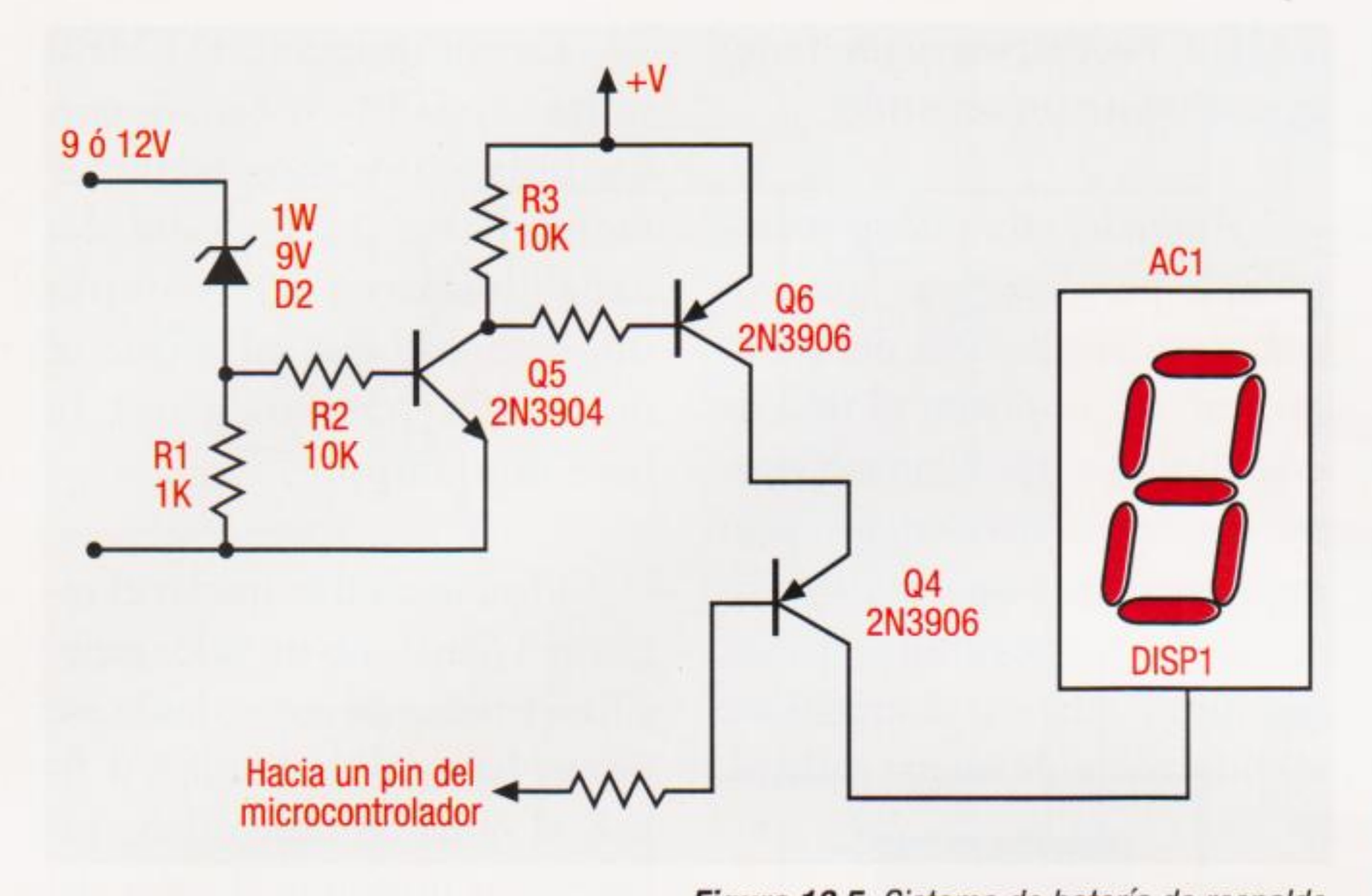


Figura 13.5 *Sistema de batería de respaldo*

el cual puede ser muy útil en el caso de que por ejemplo se interrumpa el fluido eléctrico o apague el automóvil.

Al ocurrir uno de estos eventos, la betría entra a funcionar gracias a la configuración con el diodo (D1) que se muestra en la **figura 13.5**.

En este circuito puede observar que mientras la entrada principal de alimentación de 12V se encuentra presente, el diodo no conducirá.

En el momento en que la batería o la fuente de 12V se caiga por debajo de 9VCC o sea desconectada, la batería de 9V polarizará directamente al diodo, y éste conducirá acoplando los 9V disminuidos en 0.7V de polarización del diodo a la entrada del regulador de voltaje de +5V, evitando que el microcontrolador pierda su alimentación y por lo tanto la hora exacta se vea alterada.

Para evitar que la batería de respaldo de 9V se descargue rápidamente, los *displays* que consumen la mayor parte de la corriente pedida por el circuito, deben ser desconectados en el momento en el cual ésta entra en funcionamiento.

Usando un transistor totalizador PNP (Q6) como interruptor, se puede controlar el suministro de corriente hacia los cuatro transistores (Q1 a Q4) que manejan los *displays*; para ello se ha realizado un circuito detector del voltaje de alimentación con base en un detector que usa un diodo Zener de 9V (D2) y le permite determinar cuando la alimentación está por encima o por debajo de 9V. Bajo condiciones normales (con la alimentación de 12V) el diodo Zener deja pasar 3V hacia la base del transistor NPN (Q5), los cuales son suficientes para saturarlo y ocasionar el cierre del transistor totalizador PNP que deja pasar toda la corriente que requieren los *displays*; de lo contrario (cuando esta sólo la batería de 9V) el circuito detector polariza con 0V la base de Q5, ocasionando la apertura del transistor totalizador evitando que los *displays* se enciendan mientras que el microcontrolador y el resto del circuito del reloj, conservan su energía y siguen funcionando.

Descripción del programa

El PIC16F84 posee un módulo llamado temporizador0 (Timer0), el cual es un contador/temporizador de 8 *bits*, susceptible de leerse y escribirse por *software*. Para este programa se ha aprovechado dicho módulo para generar la unidad básica de tiempo (un segundo).

Para ajustar un tiempo exacto se puede variar el tipo de cristal externo y la división de frecuencia hecha con el prescalador, tomando como entrada los ciclos internos de reloj en lo que se denomina modo de temporización, el cual se selecciona colocando en cero el *bit* 5 del registro **OPCION**.

El prescalador del PIC16F84 ha sido asignado a TMR0 para que desarrolle una división 1:16. El programa ha sido escrito para trabajar con un cristal de 1.8432 MHz, generando una frecuencia interna de 460,8 KHz; recordemos que la frecuencia del reloj se divide internamente por cuatro, generando cuatro ciclos en cua-

```
;*********************************
;CURSO PRACTICO SOBRE
MICROCONTROLADORES   -CEKIT
;PROYECTO: RELOJ DIGITAL CON PIC
;AUTOR: JUAN ANDRES CASTAÑO
; PEREIRA - COLOMBIA -2001
;*********************************
      LIST    P = 16F84

      LIST   F = INHX8M

ZEROT         equ    H'81'
UNO           equ    H'F3'
DOS           equ    H'49'
TRES          equ    H'61'
CUATRO        equ    H'33'
CINCO         equ    H'25'
SEIS          equ    H'05'
              ; TABLA DE SEGMENTOS
SIETE         equ    H'F1'
OCHO          equ    H'01'
NUEVE         equ    H'31'
BLANC         equ    H'FF'
              ;MAPA DE VARIABLES
MAXFRAC       equ    D'31'
              ;Timer : 256-31=225,
              ;ASI OBTENEMOS 1 s
MAXSECS       equ    D'196'
              ;SEGUNDOS:256-196=60
MAXMINS       equ    D'196'
              ;MINUTOS:256-196=60
MAXHRS        equ    D'244'
              ;HORAS:256-244=12
```

Figura 13.6 *Valores de inicio de los registros de ajuste de la base de tiempo*

dratura. Internamente el contador de programa (PC) es incrementado con cada cuarto pulso, llevando cada instrucción a la memoria de programa. La rotación del Timer0 (256 pasos) y su configuración establece una frecuencia final de 225 Hz.

En la **figura 13.6** se observan los valores establecidos dentro del programa para ajustar los segundos, los minutos y las horas, al igual que los valores que van a salir por el puerto B hacia los *displays* para cada uno de los valores posibles.

Una variable específica (sec_nth) se utiliza para contar los 225 desbordamientos del TMR0 necesarios para tener exactamente un segundo.

Además, ella puede manipularse para asegurar una correcta sincronización del tiempo real, sin importar el tipo de cristal que se use. Con los ajustes que se hicieron en este programa se llega a un error aproximado de 1.2 segundos al año, siempre y cuando el cristal sea el indicado y de buena calidad.

En este programa el TMR0 se ha iniciado en cero y con los pulsos internos se incrementa hasta que se desborda; cada desborde incrementa una variable que es la que se manipula para originar la base de tiempo.

Otra opción es iniciar el registro TMR0 con un valor específico e incrementarlo desde ese punto hasta el desborde. Al final, el objetivo único debe lograrse sin importar el valor del cristal o del prescalador.

Si hace uso de la segunda opción tenga en cuenta que al escribir en el TMR0 se pueden perder hasta dos ciclos de reloj mientras los flancos de entrada se alistan de nuevo.

En la **figura 13.7** se encuentra el diagrama de flujo global del proyecto, en el se puede

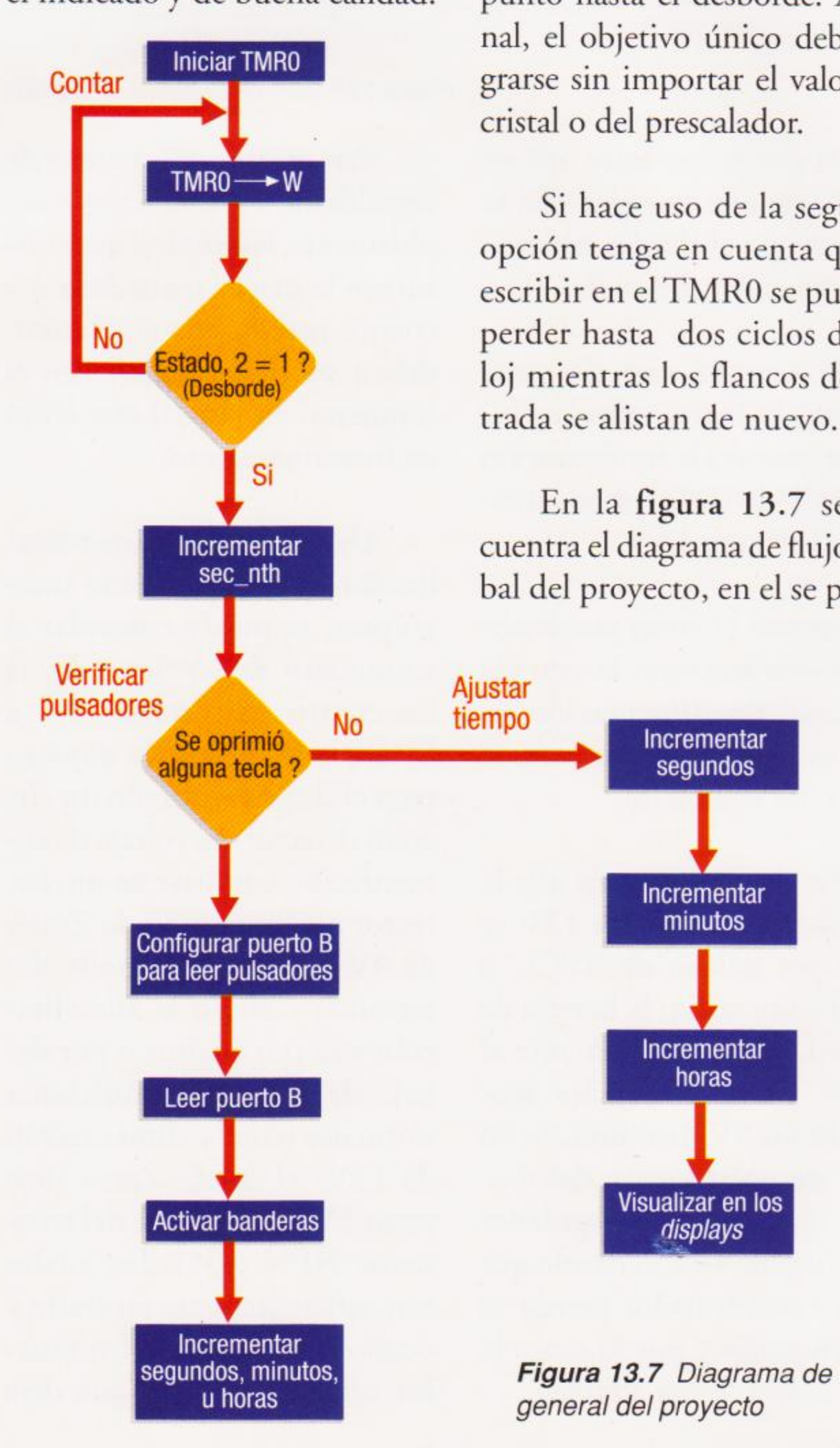


Figura 13.7 *Diagrama de flujo general del proyecto*

observar que cada desborde del TMR0 se detecta al chequear el *bit* 2 del registro de estado; para que ese *bit* se vea afectado primero se debe mover el contenido del TMR0 a W.

Cada que tiene lugar ese desborde se incrementa el registro **sec_nth** donde se almacenan las fracciones de segundo y a su vez, modifica los registros finales donde se almacenan los segundos, los minutos y las horas.

El reloj debe estar en capacidad de leer los pulsadores que configuraran su valor actual; por ello se debe incorporar una rutina que se ejecute permanentemente, en la cual se pregunta por el estado de estos pulsadores y se incrementen los valores actuales de cada uno de los registros.

```
CONTEO
        MOVF      RTCC,0
        ; RTCC -> w
        BTFSS     STATUS,Z_BIT
; EL TMR0 NO AFECTA LA CUENTA
        GOTO      CONTEO
        INCFSZ     sec_nth,1 ;
        ;INCREMENTO DE LAS FRA-
        ;CCIONES DE SEGUNDO
        GOTO      TEMPS_OK
        clrwdt
        MOVLW     MAXFRAC
        MOVWF      sec_nth
        ; RECARGA DE LAS FRACCIONES
        ;DE SEGUNDO PARA LA
        ;SIGUIENTE OCASION
VERIF_SW
        BTFSS     BANDERAS,SW_ON
        ;SE HA PRESIONADO UN PULSADOR?
        GOTO      MOSTRAR_LA_HORA
        ; NO -> MOSTRAR LA HORA
        BTFSC     BANDERAS,SW1
        ;  ES SW1 (SEGUNDOS)?
        GOTO      MOSTRAR_LA_HORA
        ; SI-> SE CAMBIA LA HORA
        MOVLW     MAXSECS
        ; REINICIAR SEGUNDOS EN
        ;CERO, CUANDO SE AJUSTE EL
        ;RELOJ
        MOVWF     SEGUNDOS
        MOVLW     H'7F'
        ; ACELERAR EL TIMER 1/2 S
        MOVWF     sec_nth
        BTFSS     BANDERAS,SW2
        ; ES SW2 (PULSADOR DE LOS
        ;MINUTOS)?
        GOTO      AJU_HORA        ;
        ;NO, NO CAMBIAR MINUTOS Y
        ;VERIFICAR LAS HORAS
        MOVLW     H'AF'
        ; SI, ACELERAR EL TIMER
        ;PARA INCREMENTAR LOS MIN.
        MOVWF     sec_nth
        INCFSZ     MINUTOS,1
        ; INCREMENTAR LOS MINUTOS
        GOTO      AJU_HORA
        ;SI SE DESBORDAN LOS MINUTOS,
        MOVLW     MAXMINS
        ; REINICIALIZACION DE LOS
        ; MINUTOS
        MOVWF     MINUTOS
```

Figura 13.8 *Incremento debido a los desbordes del TMR0 y lectura de pulsadores*

```
AJUST_TIEMPO
        BSF     BANDERAS,SEC
        ;VERIF. SI SE DEBEN ACTUA-
        ;LIZAR LOS SEGUNDOS
        BSF     BANDERAS,CHG
        INCFSZ SEGUNDOS,1     ;
        ;INCREMENTO DE LOS SEGUDOS EN 1
        GOTO    TEMPS_OK
        MOVLW  MAXSECS        ;
        ;DESBORDE-> REINICIAR SEG.
        MOVWF  SEGUNDOS
        BSF     BANDERAS,MIN   ; LOS
        ;MINUTOS DEBEN ACTUALIZARSE?
        BSF     BANDERAS,CHG  ;
        ; DEPENDE DEL CAMBIO EN LA
        ; BANDERA
        MOVLW  ADJMIN
        subwf  sec_nth,1
        ; AJUSTE PARA LA PRECISION
        INCFSZ MINUTOS,1      ;
        ;INCREMENTO DE LOS MINUTOS
        GOTO    TEMPS_OK
        MOVLW  MAXMINS
        ; REINICIALIZACION DE LOS
        ; MINUTOS
        MOVWF  MINUTOS
        BSF     BANDERAS,HRS  ; SE
        ;DEBEN ACTUALIZAR LAS
        ;HORAS?
        BSF     BANDERAS,CHG
        MOVLW  ADJHR
        ADDWF  sec_nth,1
        ; AJUSTE PARA LA PRECISION
        ;DE LAS HORAS
        INCFSZ HORAS,1        ;
        ;INCREMENTO DE LAS HORAS
        GOTO    TEMPS_OK
        MOVLW  MAXHRS         ;
        ;REINICIALIZACION DE LAS
        ;HORAS
        MOVWF  HORAS
        MOVLW  ADJDAY
        SUBWF  sec_nth,1
; AJUSTE PARA CADA MEDIODIA
(12h00)
```

Figura 13.9 *Rutina de ajuste de tiempo en los registros de segundos, minutos y horas.*

En el código de la **figura 13.8** se muestra el incremento a la variable sec_nth, al igual que la rutina de verificación de los pulsadores.

Si no se detectó ninguna señal de los pulsadores, se entra a un modo normal de trabajo, en el cual el valor de la variable sec_nth incrementa normalmente los segundos, los minutos y las horas, para luego pasar a mostrarlos en los *displays* de siete segmentos, **figura 13.9**.

El programa tiene porciones de código que actúan como manejadores de los *displays*. Existe una variable para cada uno de los cuatro *displays*. Un valor en hexadecimal de 0 a 9 puede escribirse en estas variables y ellas se convierten para dar la salida correcta hacia los *displays*. Sólo un *display* actúa en cada instante de tiempo, por lo que el programa proporciona el código correcto para cada turno.

Otra sección del código toma los valores de los segundos, minutos y horas, y los separa en los dígitos necesarios para cada *display*. En otras palabras, por ejemplo, 48 segundos se separarán en un «4» y un «8» y se escribirán en la variable apropiada de *display*.

La velocidad de ejecución de los PIC16F84 (y del resto de la familia PICmicro™) permite elaborar una gran cantidad de funciones con muy pocos pines, al multiplexar estos últimos por programa.

Ensamblaje del proyecto

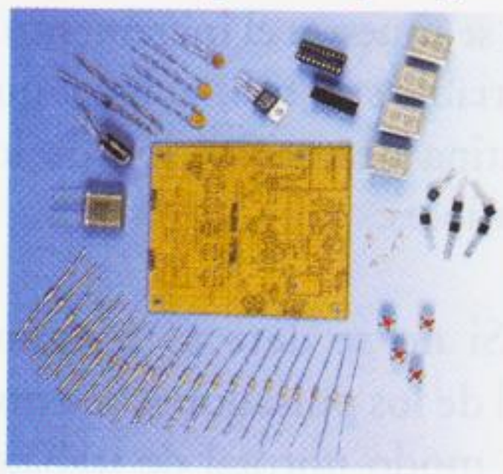


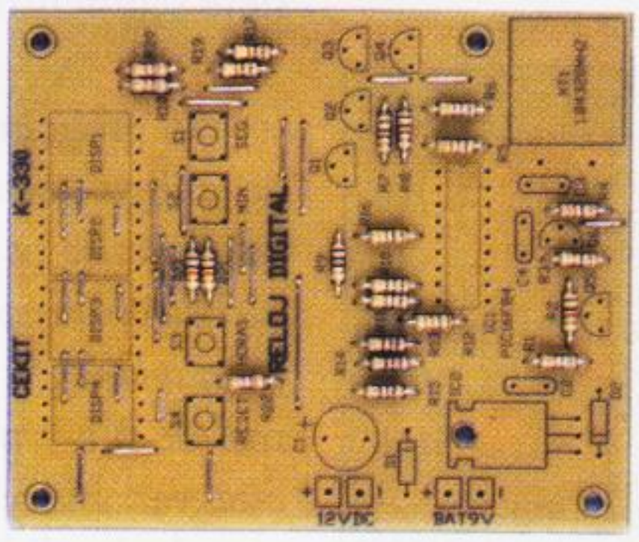
Figura 13.10 *Conjunto de componentes electrónicos y accesorios que se requieren para ensamblar el reloj digital con PIC.*

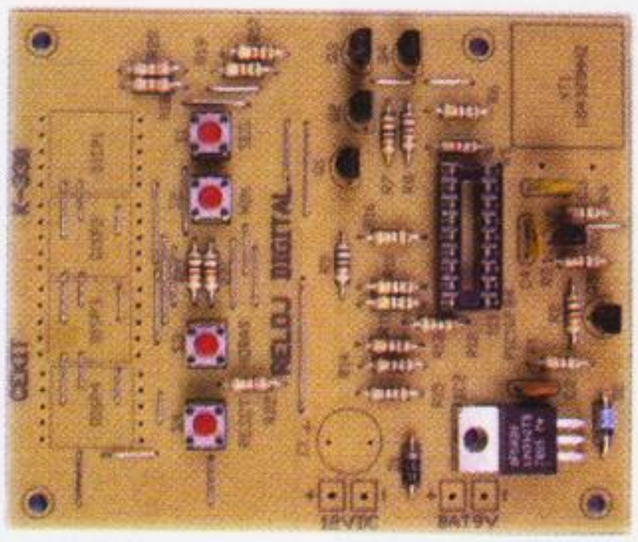
En esta ocasión se podrá construir un reloj digital con base en el microcontrolador PIC de alta precisión, que puede ser alimentado con baterias o fuentes CC entre 9 y 12 V. Por su diseño compacto puede instalarse en varios lugares tales como los vehículos, habitaciones o escritorios. Para su montaje se requieren los siguientes pasos

LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F84 programado
1	IC2	Regulador fijo de +5V -1A, 7805
5	Q1,Q2,Q3,Q4 y Q6	Transistores de propósito general PNP, 2N3906
1	Q5	Transistor de propósito general NPN, 2N3904
1	D1	Diodo de propósito general 1N4004
1	D2	Diodo zener de 9V - 1W
4	DISP1 a DISP4	Displays de siete segmentos de ánodo común
1	C1	Condensador electrolítico de 470uF - 25V
1	C2	Condensador cerámico de 0.1uF - 50V
2	C3 y C4	Condensadores cerámicos de 22pF - 50V
1	XTAL	Cristal de cuarzo de 1.84320MHz
6	R1 y R4 a R8	Resistencias de 1k - 1/4W
6	R2,R3,R17,R19,R21 y R23	Resistencias de 10k - 1/4W
8	R9 a R16	Resistencias de 100 - 1/4W
3	R18,R20 y R22	Resistencias de 820 - 1/4W
4	S1 a S4	Pulsadores pequeños de 4 pines, para circuito impreso
1	K-330	Circuito impreso CEKIT
1		Base para circuito integrado de 18 pines
4		Espadines

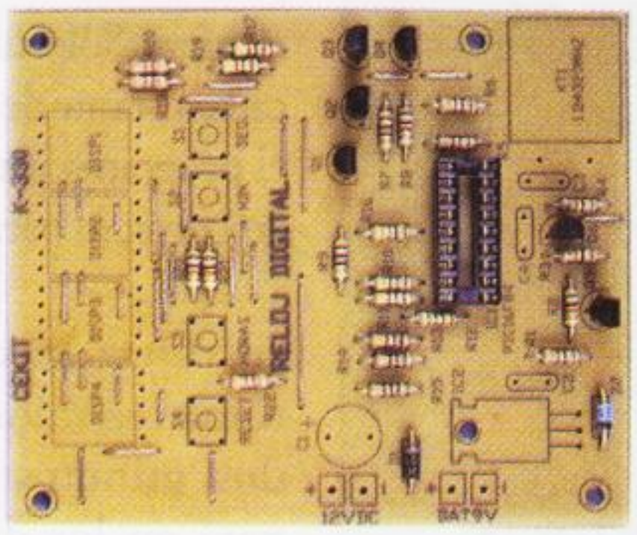
Paso 1
Instale y suelde los puentes de alambre y las resistencias R1 a R22.
Figura 13.11

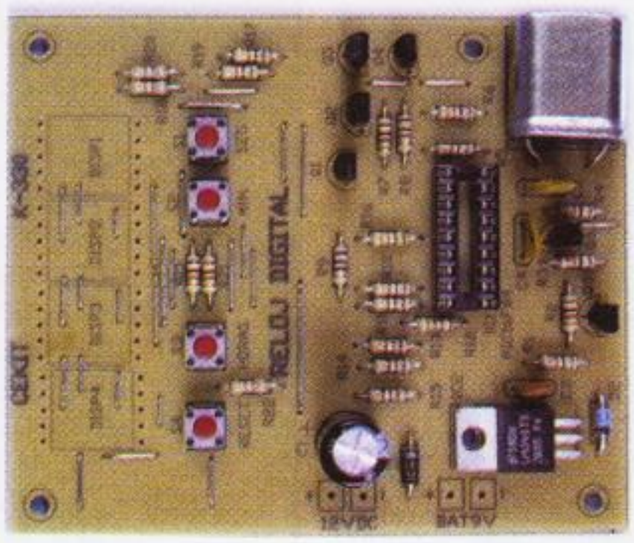




Paso 4
Instale y suelde los condensadores cerámicos C2, C3 y C4.
Figura 13.14

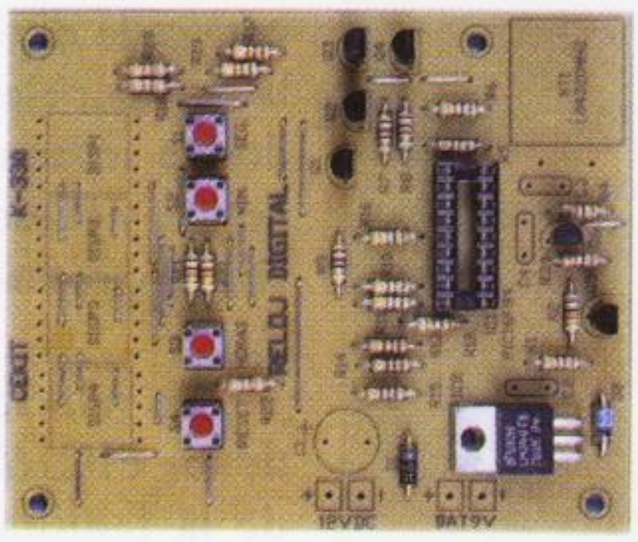
Paso 2
Luego instale y suelde la base para el circuito integrado IC1, los transistores Q1 a Q6 y los diodos rectificador y Zener D1 y D2 .
Figura 13.12

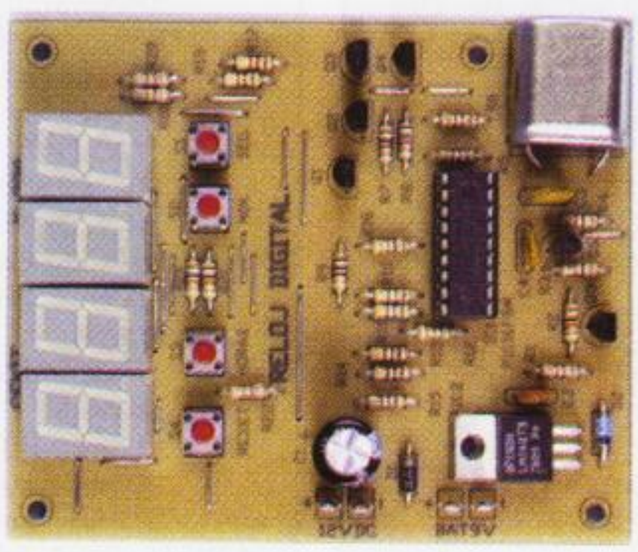




Paso 5
Fije y suelde en posición axial (horizontal) y en el lugar indicado el cristal de cuarzo XT1 y suelde también el condensador electrolítico C1.
Figura 13.15

Paso 3
Instale y suelde el circuito integrado regulador de voltaje IC2 y los pulsadores pequeños S1 - S4.
Figura 13.13

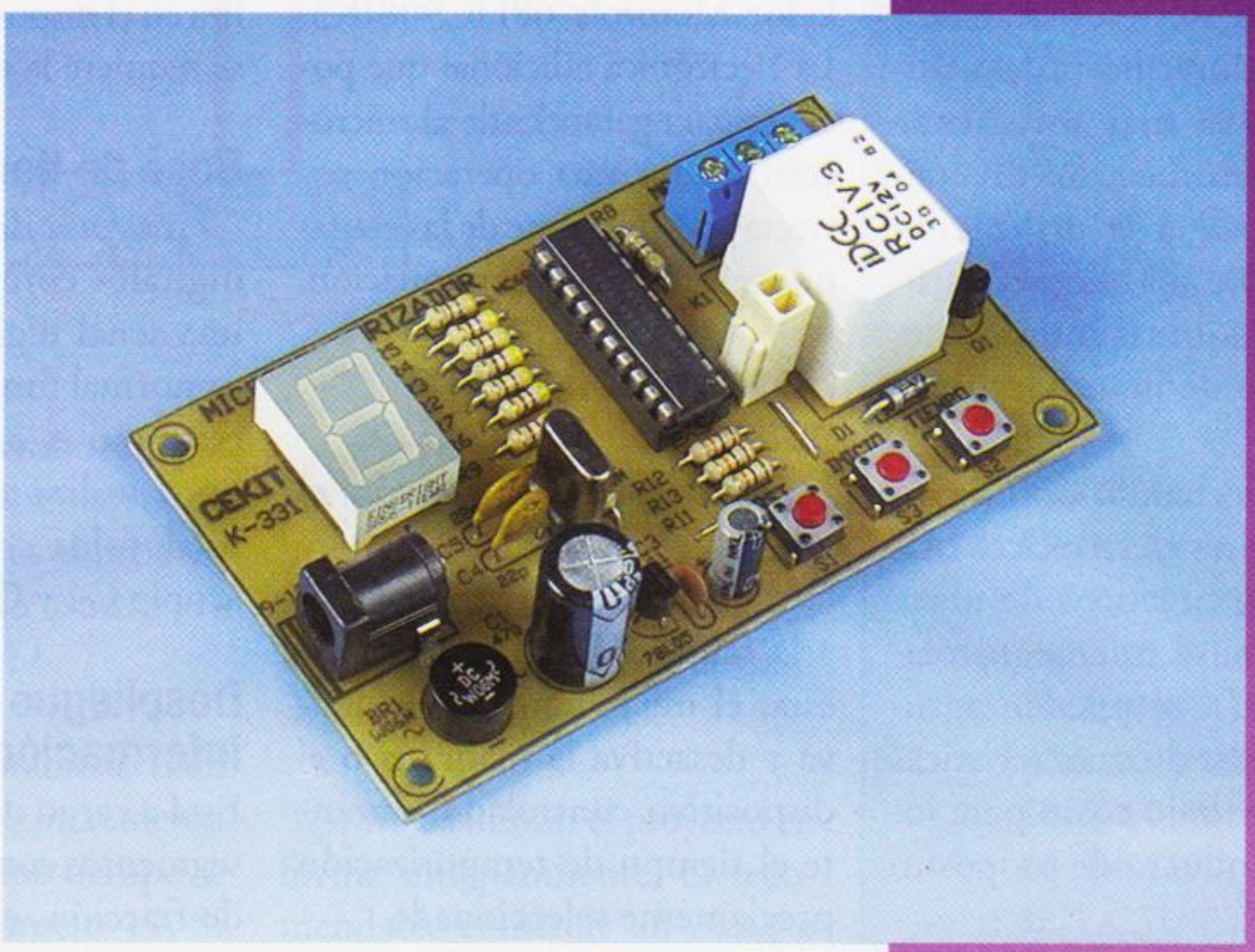




Paso 6
Aspecto final del reloj digital ensamblado, luego de instalar y soldar los displays y los espadines para las entradas de alimentación principal (12 V) y de respaldo (9 V).
Figura 13.16

Paso 7. Calibración del reloj
Cada vez que se alimente el circuito o se oprima el pulsador de reset, en los displays se debe visualizar el número 12.00, que es el dato con que siempre inicia su operación. Ahora debe ajustar el reloj con respecto a la hora actual; oprimiendo el botón de horas y minutos usted logra un incremento progresivo de los datos actuales, hasta llegar a los números deseados. Si oprime el boton de segundos, en los displays desaparece la hora y solo se activan dos displays indicando los segundos que corren actualmente.

PROYECTO 14



MICROTEMPORIZADOR

Construya este sencillo temporizador e incorpórelo en diversos proyectos que requieran control sobre cortos intervalos de tiempo.

Características

- Gran exactitud en el tiempo
- Tamaño reducido, con adecuado despliegue de información y manejo
- Inmunidad al ruido, muy importante en ambientes industriales
- Capacidad para manejo de potencia
- Bajo costo
- Consumo: 0.6W
- Escalas de Tiempo: 0.5, 1.0, ..., 9.5 s
- Despliegue del tiempo seleccionado
- Pulsador para fijar el tiempo

Los temporizadores son uno de los circuitos más utilizados en la electrónica. Existen múltiples alternativas para construirlos, desde una simple red RC (resistencia + condensador), hasta la utilización de la más sofisticada microelectrónica. Su diversidad obedece a la utilización que se hace de ellos en productos, máquinas industriales y procesos de manufactura.

Aprovechando uno de los componentes electrónicos más versátiles y prácticos de nuestro tiempo, el microcontrolador (MCU), se pueden agrupar una serie de características técnicas de bajo costo para lograr un producto de propósito general.

El temporizador descrito en este proyecto es, en esencia, un dispositivo que se activa al detectar el cambio de estado de un sensor y "espera" un determinado tiempo, al cabo del cual efectúa algún tipo de acción. El puede ser utilizado en procesos tales como:

- Sellado de bolsas plásticas y materiales de empaque.
- Soldadura de punto.
- Exposición de películas y planchas en procesos de *screen* y estampado.
- Activación de alarmas.
- En general, en máquinas que requieran temporización con tiempos cortos y de buena precisión.

Descripción del circuito

Observe, en la **figura 14.1**, el diagrama esquemático del temporizador, constituido en gran parte por un microcontrolador Motorola 68HC908JK3. La electrónica adicional que posee es para garantizarle al microcontrolador una operación correcta y se compone de las etapas que se describen a continuación.

Etapa de salida de potencia

Hacen parte de esta sección: el relé K1, el transistor 2N3904 y la resistencia R10, conectados a la salida PTD6. A través de ésta, el microcontrolador activa y desactiva la máquina o el dispositivo controlado, durante el tiempo de temporización previamente seleccionado.

Si su proyecto requiere de una mayor potencia, puede utilizar la salida del relé K1 para manejar un relé de más corriente o un contactor.

También se pueden sustituir el relé y el transistor por una pareja de optoacoplador y Triac, este último seleccionado con la potencia suficiente para manejar la carga y montado sobre un disipador de calor de tamaño adecuado. Para cargas CC, podría utilizarse un MOSFET de potencia.

Etapa de entradas

El pulsador S2, conectado a la entrada PTD5, es el elemento que permite seleccionar el tiempo durante el cual la acción se va a ejecutar, mientras que la activación de la entrada (PTD4) puede provenir de diferentes fuentes, tales como microinterruptores, pedales, pulsadores, etc.; cualquiera de ellos se localiza en el dispositivo sobre el cual se requiere la temporización.

Base de tiempo

La mayoría de las aplicaciones digitales complejas requieren una señal digital de reloj para su normal funcionamiento. En este caso el oscilador es interno, y utiliza para ello el cristal XT1 y los condensadores de acople C4 y C5.

Despliegue de información

Está a cargo del *display* de siete segmentos con punto indicador de fracción, el cual muestra el último tiempo que se ha seleccionado a través del pulsador S2. Adicionalmente, se requiere de una resistencia por cada barra indicadora del *display* con el fin de asegurar una luminosidad uniforme. Éstas corresponden dentro del circuito a R2, R3, R4, R5, R6, R7, R8 y R9, para los segmentos a, b, c, d, e, f, g y pd, respectivamente.

Funcionamiento

Para establecer el tiempo presione el pulsador S2 hasta que se obtenga el valor deseado en el *display*; el punto decimal indica 0.5 s más que el dígito a la izquierda. La **figura 14.2** muestra como se interpreta el tiempo en el *display*. Este ajuste sólo debe efectuarse una vez, a no ser que desee cambiar el tiempo.

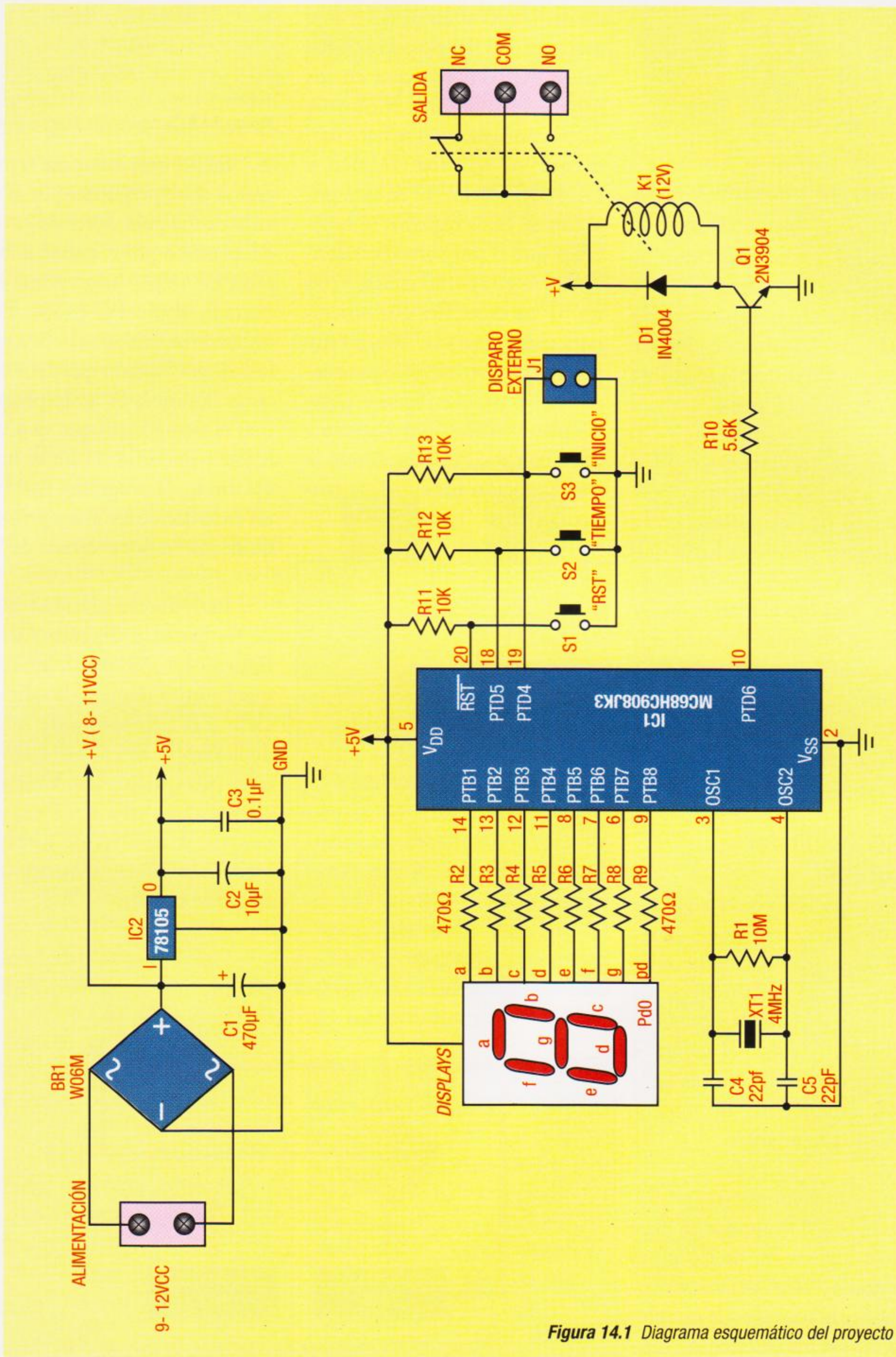


Figura 14.1 *Diagrama esquemático del proyecto*

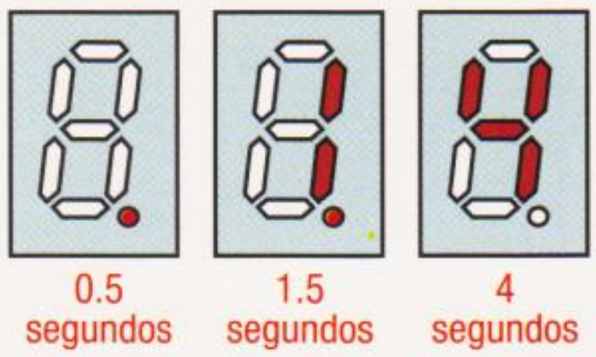


Figura 14.2 *Interpretación del display*

Una vez conectado el temporizador accione el "mando externo", que por lo general es un mecanismo o dispositivo de la máquina que desea controlar (microinterruptor, pulsador, contacto de relé, pedal, etc).

En este instante el temporizador activa su relé interno, permitiendo que el proceso en la máquina se produzca hasta que finalice el período de temporización. Para que el temporizador actúe de nuevo hay que liberar el "mando externo" y volverlo a accionar, si éste quedase activado. La anterior condición es conveniente en la mayoría de las aplicaciones industriales.

Descripción del programa

El primer paso, que se sigue en este programa, es asignar un registro específico para almacenar el dato actual de la temporización, este registro se ha llamado TEMP y sobre él vamos a efectuar una serie de operaciones de incremento, decremento y llamado a tablas.

Como siempre, en la primera sección del programa se hace una configuración de los pines de entrada y salida del microcontrolador, de acuerdo a lo establecido en los parámetros de funcionamiento y en el diagrama esquemático del proyecto, **figura 14.3**. Recuerde que en los microcontroladores Motorola dicha configuración se hace mediante los registros DDRB y DDRD; con un "1" en una posición de estos registros se especifica que el pin correspondiente funcionará como salida, y con un "0" se especifica que funcionará como entrada. Tenga muy presente este hecho porque puede equivocarse fácilmente, especialmente si está muy familiarizado con los PIC, en donde la configuración funciona de manera inversa.

```
INICIO:
        RSP
        CLRA
        CLRX
        BSET
        COPD,CONFIG1 ;
         ; DESHABILITO EL
         ; COP (WATCHDOG)
        MOV #$FF,DDRB ;
         ;CONFIGURO EL PUER-
         ;TO B COMO SALIDAS
        CLR PORTB
        MOV #$C0,DDRD ;
         ;CONFIGURO EL PUER-
```

```
         ;TO D PARA LEER DATOS
          CLR PORTD
         ; POR PD4 Y PD5
```

Figura 14.3 *Configuración de los pines I/O*

En primera instancia, el programa debe preguntar por el estado de los pulsadores de entrada, primero por el conectado al pin PD5 (S2) y luego por el conectado al pin PD4 (S3). Este orden de prioridad se debe a que la primera función que debe ejecutar la tarjeta es la de programar el tiempo con que va a trabajar, y mientras S3 no se haya oprimido, el programa entra en un ciclo donde retorna a preguntar de nuevo por el estado de S2.

Para mayor claridad puede remitirse al diagrama de la **figura 14.4**. Allí puede apreciar que la única manera de salir del ciclo e ir a la rutina de inicio de la temporización es cuando en el pin 4 del puerto D aparece un cero; mientras eso no ocurra, se retorna de nuevo a leer S2 para seguir incrementando al registro TEMP.

Cuando se sale de la rutina anterior, el registro TEMP

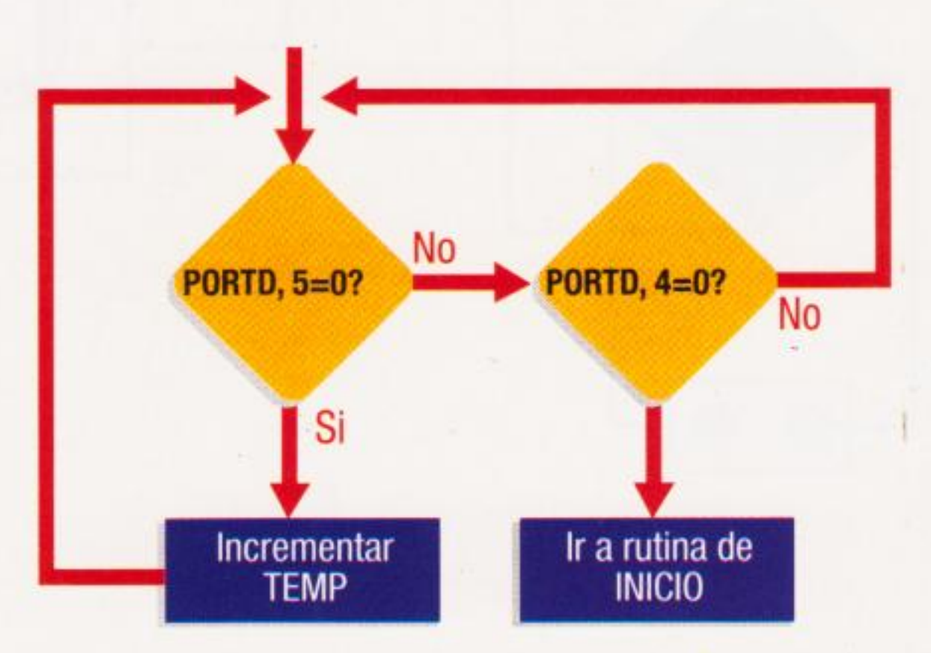


Figura 14.4 *Ciclo de inicio de chequeo de los pulsadores*

ha quedado cargado con un valor de temporización. Dependiendo de su valor, se debe originar la salida hacia el *display* de siete segmentos. Si observa cuidadosamente el diagrama esquemático notará que el punto decimal (que es parte fundamental en el funcionamiento de este proyecto) está conectado a un pin del puerto D, mientras todos los demás segmentos están conectados al puerto B.

Para resolver esta situación existen varias opciones, en este caso se decidió construir dos tablas distintas pero de la misma dimensión, cada una con el dato que se va a enviar por uno de los puertos, **figura 14.5**.

```
**********************
* UBICACIÓN DE TABLAS EN *
* MEMORIA *
*    MEL y MEL1      *
**********************
        ORG  MEL
        FCB  UNO , DOS ,
TRES , CUATRO , CINCO ,
SEIS , SIETE , OCHO ,
NUEVE, DIEZ
        FCB  ONCE , DOCE
, TRECE, CATORCE, QUINCE,
DSEIS, DSIETE , DOCHO ,
DNUEVE
        ORG  MEL1
      FCB  UNOD, DOSD,
TRESD, CUATROD, CINCOD,
SEISD, SIETED, OCHOD,
NUEVED, DIEZD
        FCB  ONCED ,
DOCED , TRECED , CATORCED
, QUINCED ,
DSEISD,DSIETED,DOCHOD,
DNUEVED
```

Figura 14.5 *Tablas con los datos para cada uno de los puertos*

La idea es direccionar mediante el registro TEMP la misma dirección en ambas tablas; el dato de la primera se envía al puerto B y el dato rescatado de la segunda se envía al puerto D, **figura 14.6**.

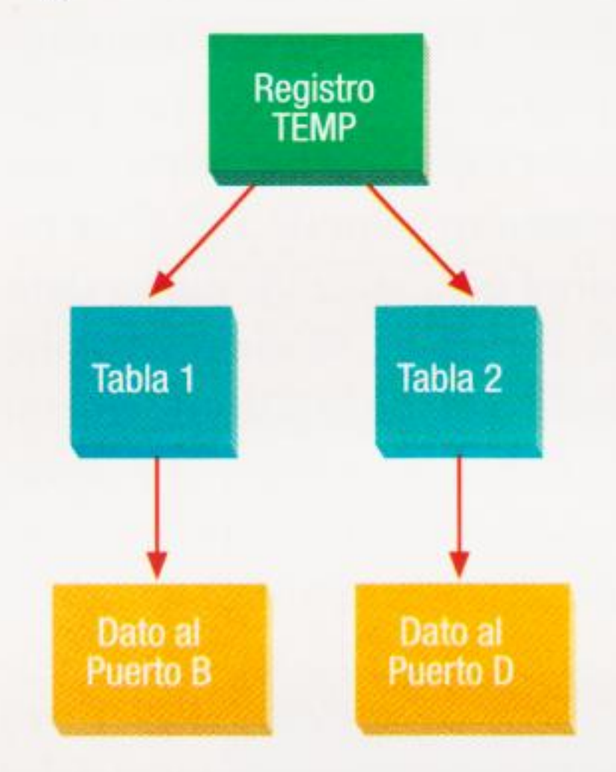


Figura 14.6 *Manejo de las tablas para formar los segmentos*

Los valores que poseen cada uno de los elementos de esta tabla ya se han definido como constantes en el inicio del programa, **figura 14.7**. Tenga en cuenta que el punto decimal configura un valor diferente para cada uno de los decimales del *display*, llegando hasta 19 valores distintos.

```
; CURSO PRÁCTICO SOBRE
MICROCONTROLADORES
; FASCÍCULO 16
; MICROTEMPORIZADOR

$Include 'jl3regsg.inc'
RAMStart      EQU  $0080
RomStart      EQU  $EC00
;
;Vectores      EQU  $FFDE
RomFin        EQU  $FBFF
RESET_VEC     EQU  $FFFE
;Reset Vector
      org RamStart
CERO          EQU $7E ; 0
CEROD         EQU $00
UNO           EQU $7E
UNOD          EQU $01
DOS           EQU $0C ; 1
DOSD          EQU $00
TRES          EQU $0C
TRESD         EQU $01
CUATRO        EQU $B6 ; 2
CUATROD       EQU $00
CINCO         EQU $B6
CINCOD        EQU $01
SEIS          EQU $9E ;3
SEISD         EQU $00
SIETE         EQU $9E
SIETED        EQU $01
OCHO          EQU $CC ;4
OCHOD         EQU $00
NUEVE         EQU $CC
NUEVED        EQU $01
DIEZ          EQU $DA ;5
DIEZD         EQU $00
ONCE          EQU $DA
ONCED         EQU $01
DOCE          EQU $FA ;6
DOCED         EQU $00
TRECE         EQU $FA
TRECED        EQU $01
CATORCE       EQU $0E ;7
CATORCED      EQU $00
QUINCE        EQU $0E
QUINCDED      EQU $01
DSEIS         EQU $FE  ;8
DSEISD        EQU $00
DSIETE        EQU $FE
DSIETED       EQU $01
DOCHO         EQU $CE ;9
DOCHOD        EQU $00
DNUEVE        EQU $CE
DNUEVED       EQU $01
```

Figura 14.7 *Definición de las constantes para cada uno de los segmentos*

Cuando salimos del primer ciclo de lectura de los pulsadores entramos a una rutina de inicio de temporización, la cual se aprecia en la **figura 14.8**. En ella puede ver claramente como a través del registro acumulador (A) se direccionan y rescatan los distintos valores de acuerdo al valor actual de TEMP. Se toma el primer valor de la tabla 1 y se envía al puerto B; acto seguido, se direcciona y se toma el valor de la **tabla 14.2** y se envía al puerto D.

Entre estos dos valores configuran la visualización deseada en el *display* de siete segmentos.

Cuando el dato definitivo se encuentra en los puertos del microcontrolador, éste debe permanecer allí durante un tiempo determinado y para ello se invoca una rutina de retardo, **figura 14.9**. En esta rutina se acude al ya conocido método de ejecutar ciclos anidados hasta llegar a un tiempo determinado.

Cuando el retardo ha culminado su ejecución, se decrementa en una unidad el registro TEMP; así vamos disminuyendo los valores del temporizador cada cierto tiempo y con ese nuevo valor de TEMP se retorna para sacar un nuevo dato de las tablas. El ciclo se repite hasta que TEMP llegue a cero, cuando esto ocurra, el programa debe retornar al inicio, dando final a la temporización.

Existen algunos factores de riesgo sobre los que se debe prestar mucha atención para que el sistema funcione correctamente. Si observa la figura 14.4 puede ver que si el pulsador S2 no se ha accionado en ninguna ocasión, y si se pulsa S3 el sistema tenderá a temporizar cero segundos. De la misma manera hay que tener en cuenta que no se puede permitir que TEMP tome un valor superior a 19, porque rebasaría la capacidad del *display*. Es decir, el *display* no tiene como indicar, por ejemplo, el número 21. Para ello se ha implantado un registro de seguridad que sirve de testigo sobre los fenómenos que sucedan en el sistema y que sirve para evitar que ocurran situaciones indeseables. Este registro se ha llamado BANDERA, **figura 14.10**.

Cuando usted pulsa por primera vez a S2, inmediatamente se activa el *bit* 1 del registro BANDERA, indicando que ya hay un valor viable para la temporización; de manera que cada vez que de la orden de inicio por medio de S3, lo primero debe hacer el programa es preguntar por el estado

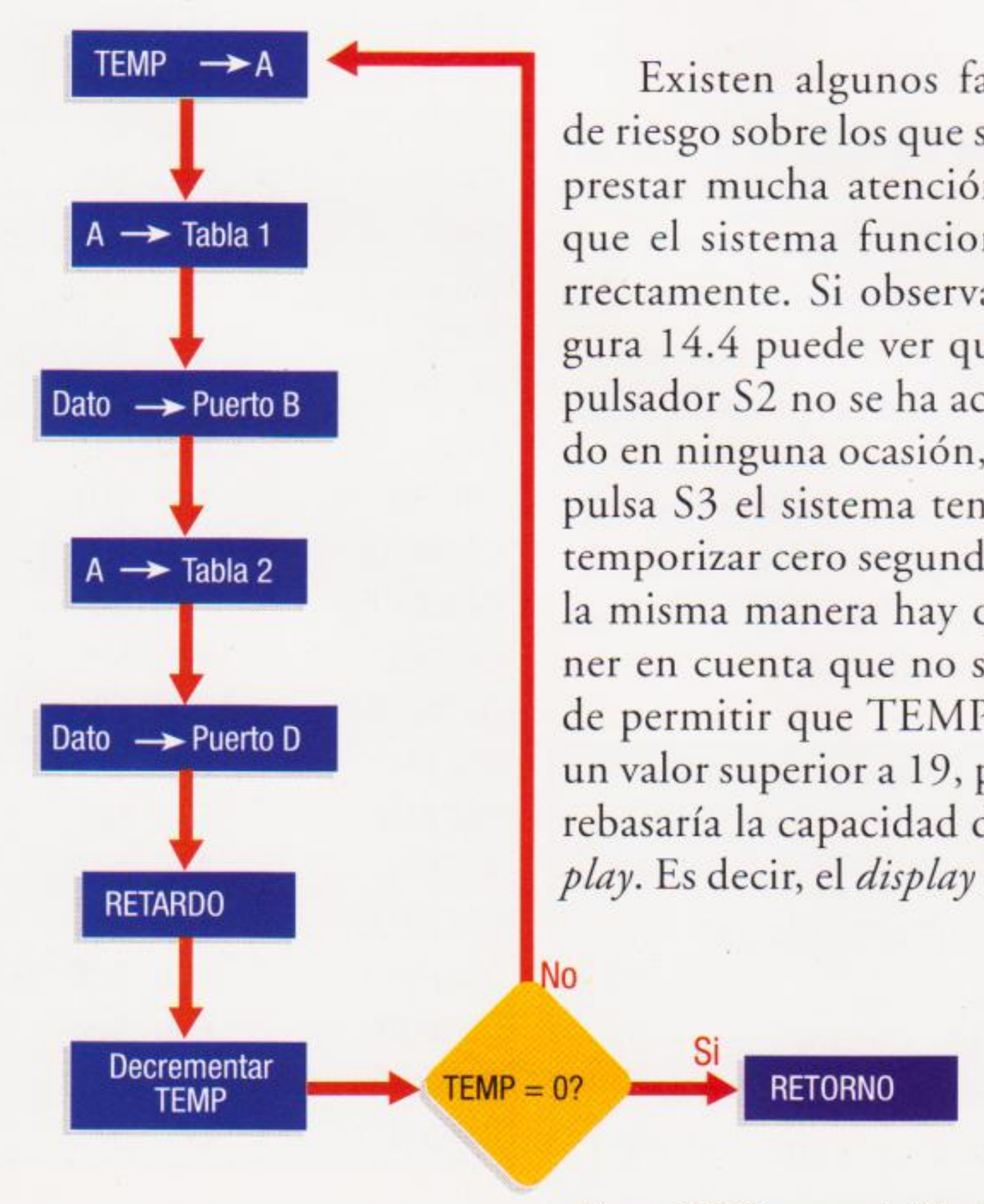


Figura 14.8 *Diagrama de flujo del cuerpo principal del programa*

```
************************
* RETARDO- SUBRUTINA DE *
* RETARDO POR SOFTWARE *
*
************************

RETARDO:
        LDA     #110T
        STA     MEM1
ET1     JSR     RETITO
        DEC     MEM1
        BNE     ET1
        RTS
RETITO:
        LDA     #228T ;
ET2     CLRX
ET3     DECX
        BNE     ET3
        DECA
        BNE     ET2
        RTS
```

Figura 14.9 *Rutina básica de retardo por software*

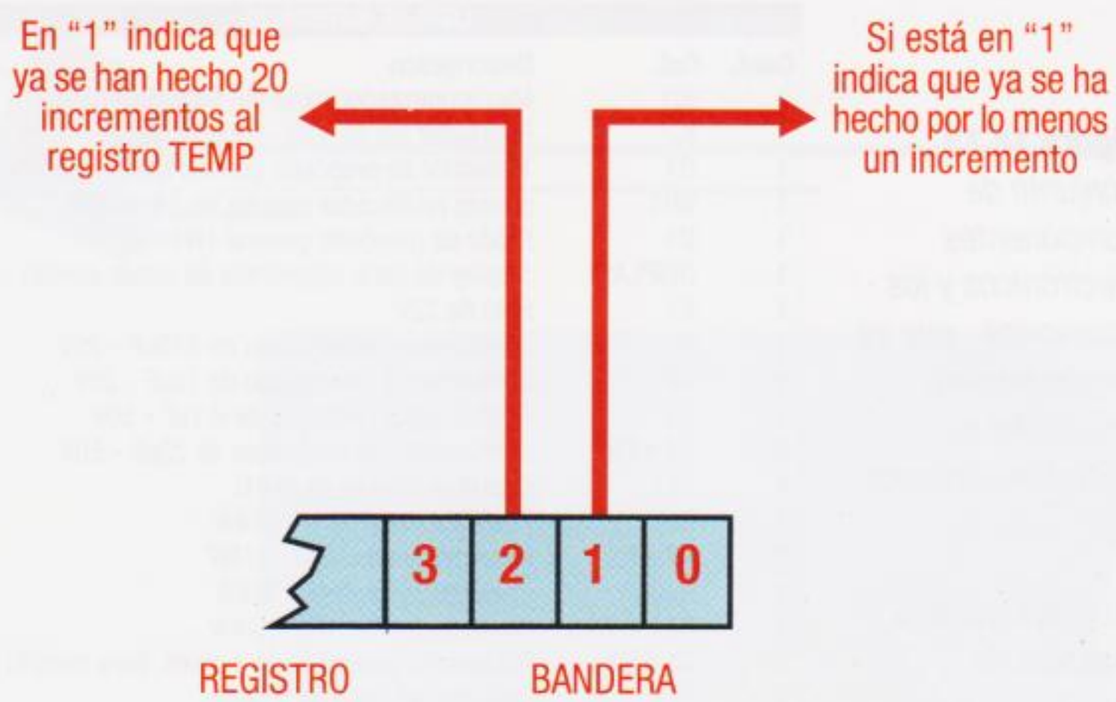


Figura 14.10 *Registro BANDERA*

de ese *bit* para corroborar que efectivamente se encuentre en alto; de lo contrario el programa retornará a su punto de inicio, **figura 14.11**.

De la misma manera, cada vez que pulse S2, se incrementa TEMP pero al mismo tiempo se compara con el número 19. Si detecta que TEMP ha sobrepasado ese valor, debe activar el *bit* 2 del registro BANDERA.

Por esa razón, antes de incrementar TEMP debido a una lectura positiva de S2, debe corroborar que dicho *bit* se encuentra desactivado; de lo contrario, TEMP no incrementará más.

En el diagrama de la **figura 14.12** se muestra el esquema general para la conexión del microtemporizador en una aplicación real. Por medio del pulsador S2 se ajusta el tiempo de la tarjeta y por medio del pulsador S3 o un mando externo (puede ser un pedal o cualquier otras entrada del usuario) disparar el funcionamiento del temporizador y la salida por medio de un relé.

Para el ensamblaje, se montan primero los puentes de alambre, las resistencias y la base para el microcontrolador. Se continúa con los condensadores, el transistor, el cristal y el relé.

Si va a utilizar el proyecto en forma independiente, es recomendable construir un *chasis* o caja para protegerlo de las condiciones ambientales. Los materiales para ésta pueden ser aluminio, hierro, plástico o acrílico; la forma de la caja, así como su tamaño, los dejamos a su gusto e imaginación.

En ciertos casos, como por ejemplo en paneles o tableros de control, el temporizador se puede montar directamente con tornillos que fijen el circuito impreso y el transformador a la base o soporte y se deja una pequeña ventana para visualizar el *display*.

```
PRINCIPAL
        CLR       TEMP
; INICIALIZÓ LA VARIABLE
DEL TIEMPO
        CLR       BANDERA
; INICIALIZÓ REGISTRO DE
BANDERAS

ET1     JSR       RETARDO
;LEER BANDERA 2 Y SI
;ESTA BRINCAR A ET2
        BRCLR     PD5,PORTD,INCREM
        ; CHEQUEO PULSADOR DE
        ; INCREMENTO DE TIEMPO
ET2     BRCLR PD4,PORTD,INICIO
        ; CHEQUEO PULSADOR
        ; DE INICIO
        ; DE LA TEMPORIZACIÓN
        BRA       ET1
INCREM:
        BSET      BANDERA,1
        INC       TEMP
        ; DECREMENTAR 10 Y
        ; AL LLEGAR PONER
        ; BANDERA 2
        BRA       ET1
```

Figura 14.11 *Activación del bit 1 del registro BANDERA cuando se ha detectado la activación deS2*

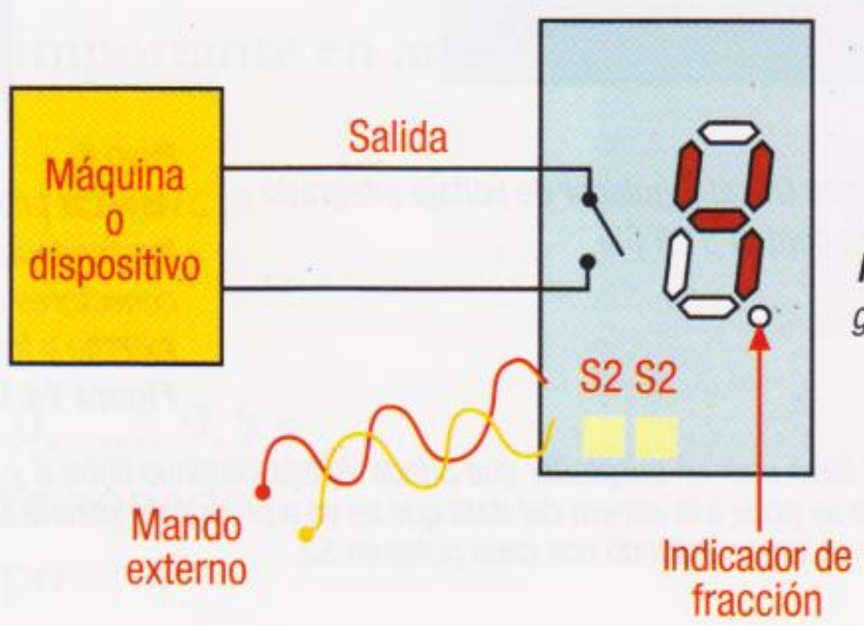


Figura 14.12 *Esquema global de la conexión de la tarjeta*

Ensamblaje del proyecto

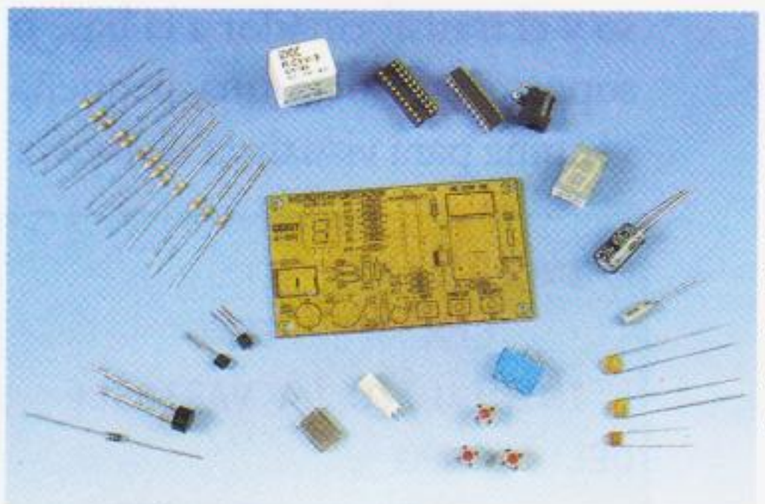
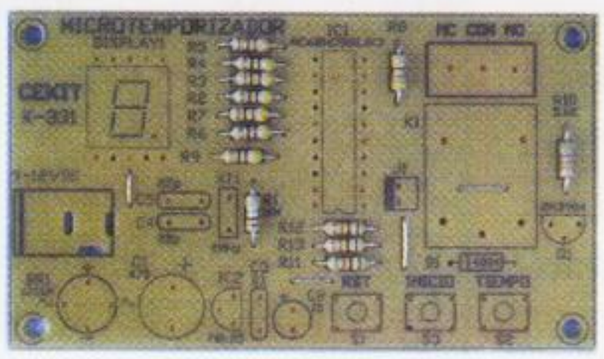
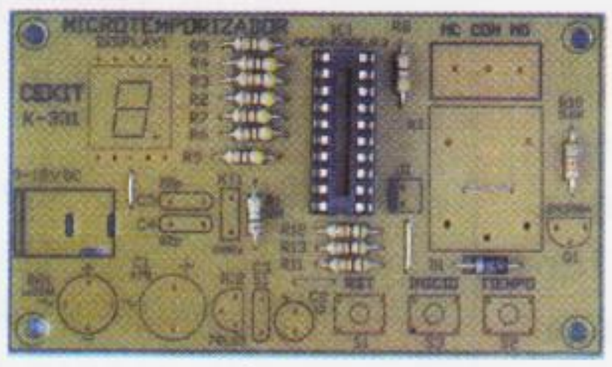


Figura 14.13
Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el microtemporizador.

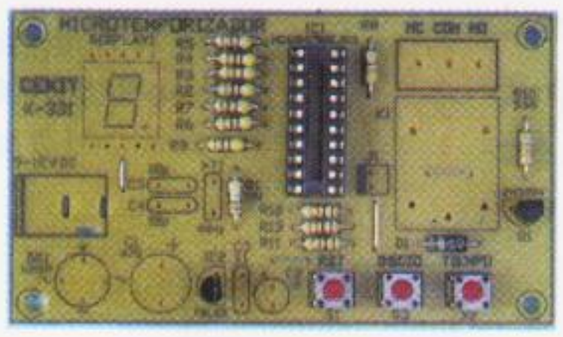
Por medio de este proyecto usted podrá programar un intervalo corto de tiempo y accionar o energizar cargas o dispositivos de hasta 10A por medio de los contactos de salida que ofrece su relé. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:



Paso 1
Instale y suelde los puentes de alambre y las trece resistencias R1 a R13.
Figura 14.14



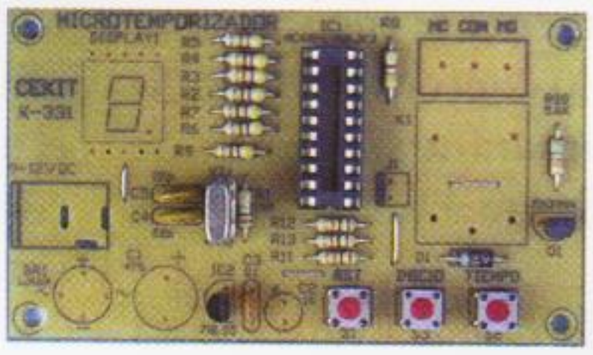
Paso 2
Luego instale y suelde la base para el circuito integrado IC1, y el diodo rectificador D1.
Figura 14.15



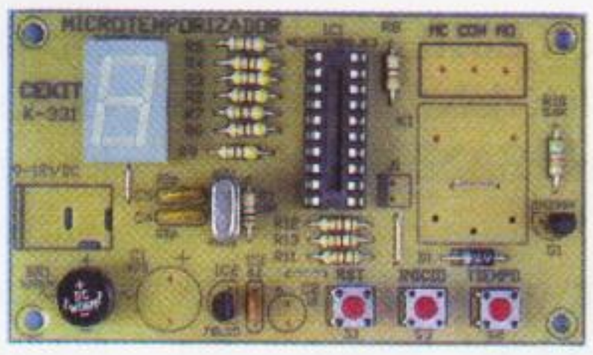
Paso 3
Instale y suelde el transistor Q1, el regulador de voltaje integrado IC2 y los pulsadores para control S1 a S3.
Figura 14.16

LISTA DE MATERIALES

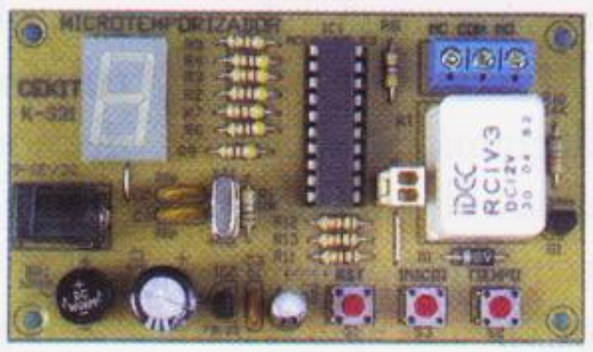
Cant.	Ref.	Descripción
1	IC1	Microcontrolador motorola MC68HC908JK3
1	IC2	Regulador fijo de +5V -100mA, 78L05
1	Q1	Transistor de propósito general NPN, 2N3904
1	BR1	puente rectificador circular de 1A, W06M
1	D1	Diodo de propósito general 1N4004
1	DISPLAY1	*Display* de siete segmentos de ánodo común
1	K1	Relé de 12V
1	C1	Condensador electrolítico de 470uF - 25V
1	C2	Condensador electrolítico de 10uF - 25V
1	C3	Condensador cerámico de 0.1uF - 50V
2	C4 y C5	Condensadores cerámicos de 22pF - 50V
1	XT1	Cristal de cuarzo de 4MHz
1	R1	Resistencia de 10M - 1/4W
8	R2 a R9	Resistencias de 470 - 1/4W
1	R10	Resistencia de 5.6k - 1/4W
3	R11 a R13	Resistencias de 10k - 1/4W
3	S1 a S3	Pulsadores pequeños de 4 pines, para circuito impreso
1	J1	Conector en línea de 2 pines
1	K-331	Circuito impreso CEKIT
1		Conector de alimentación CA/CC
1		Conector de tornillo de 3 pines
1		Base para circuito integrado de 20 pines



Paso 4
Instale y suelde los condensadores cerámicos C3, C4, y C5, y el cristal de cuarzo XT1.
Figura 14.17



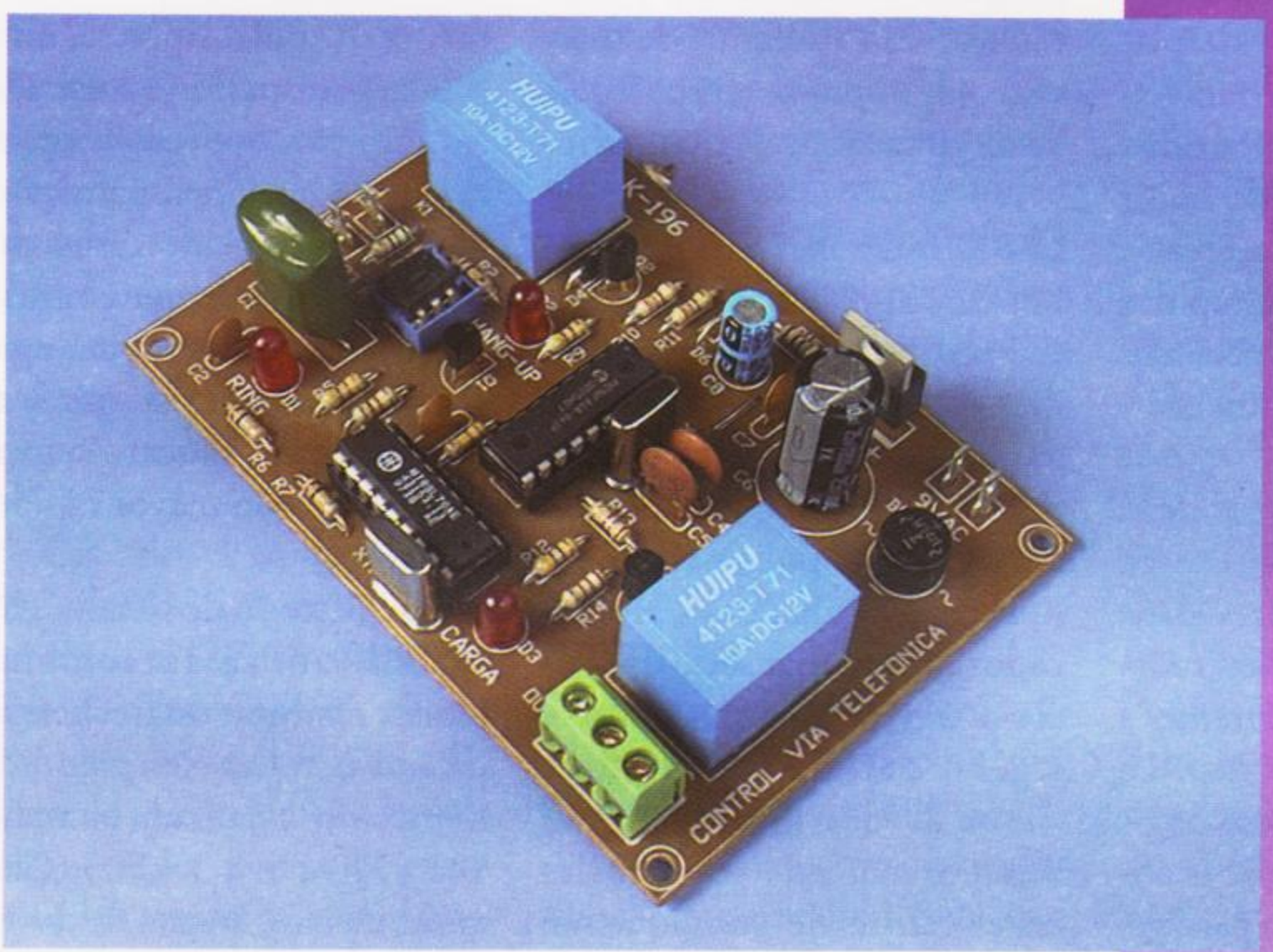
Paso 5
Fije y suelde el puente rectificador BR1 y el display de 7 segmentos de ánodo común.
Figura 14.18



Paso 6
Aspecto final del proyecto ensamblado, luego de instalar y soldar los condensadores electrolíticos C1 y C2, el relé de salida y los conectores de entrada y salida para la alimentación, el disparo externo y los contactos del relé.
Figura 14.19

Paso 7. Pruebas finales
Para alimentar el circuito debe usar un adaptador que brinde voltaje continuo entre 8 y 12 voltios. Cuando se alimenta la tarjeta por medio del conector CA/CC el circuito se pone a la espera del dato que se va a programar para la temporización. En el estado inicial el display debe mostrar el número 0 y se va incrementando con cada pulso en S2.

PROYECTO 15



CONTROL DE APARATOS VÍA TELEFÓNICA

Con este proyecto usted va a construir un equipo que le permitirá activar y desactivar una carga remota, utilizando como medio de transmisión la línea telefónica.

Características

- Activa cargas a distancia utilizando la línea telefónica
- Tamaño reducido
- Capacidad para manejo de potencia
- Bajo costo
- Salida por medio de relé
- Señalización por medio de diodos LED

El control a distancia es una modalidad que presenta muchas opciones y beneficios para los usuarios. Dentro de sus principales ventajas se encuentran: la comodidad que le ofrece al no tener que desplazarse hasta algún sitio para hacer alguna tarea, la seguridad obtenida ya que puede hacer labores peligrosas sin acercarse al área de trabajo y la rapidez con que puede tomar acciones respecto a alguna condición específica, entre otras.

Este proyecto consiste en crear un equipo que, a través de la línea telefónica, reciba órdenes o instrucciones referentes a la tarea que debe efectuar. La idea es que usted pueda, desde la comodidad de su casa u oficina, utilizar el teléfono para ejecutar alguna acción. Por ejemplo, antes de salir de su oficina puede llamar a su casa, donde tiene el aparato controlador instalado, y ordenar que se encienda el calentador de agua del baño para que cuando llegue ésta se encuentre en una temperatura ideal.

Este dispositivo está pensado para trabajar única y exclusivamente con líneas telefónicas estándar, en ningún caso con líneas integradas a conmutadores privados del tipo PBX, ya que estas pueden presentar problemas de incompatibilidad debido a que trabajan con niveles de voltaje y corriente diferentes. Adicionalmente, las órdenes de control se proporcionan utilizando el teclado del teléfono, por lo que se restringe el funcionamiento del equipo a líneas cuyo sistema de señalización o de marcado sea de tonos. Bajo ninguna circunstancia puede trabajar con teléfonos de marcado por impulsos (como los de disco giratorio, por ejemplo).

El sistema de señalización DTMF

Existen actualmente dos formas de enviar a la central de conmutación la información de los dígitos marcados en un teléfono: por pulsos y por tonos. En el primer caso, el marcador genera una serie de pulsos a través de la línea. En el segundo, el marcador produce tonos de dos frecuencias. El número marcado se identifica en el centro de conmutación contando los pulsos o decodificando los tonos, respectivamente. El método de tonos se conoce técnicamente como señalización DTMF (*dual-tone multifrequency*: multifrecuencia de doble tono).

La forma original del marcador de pulsos es la de un disco electromecánico giratorio, dotado de un sistema de levas y engranajes que gira a medida que el disco lo hace. Cuando se libera el disco, un mecanismo regulador de velocidad, a base de resortes, lo devuelve a su posición original. Los pulsos se generan durante el regreso del disco y los producen un par de contactos que se abren y cierran a razón de aproximadamente 10 pulsos por segundo (20 en Europa). Los marcadores digitales o electrónicos simulan la acción mecánica de los marcadores de disco mediante un teclado que emite los pulsos, a medida que se ingresa cada dígito. El sistema de teclado permite marcar el número deseado con mayor rapidez.

El método de señalización DTMF utiliza 16 combinaciones distintas de frecuencias de audio, todas comprendidas dentro de la llamada banda de voz (300 Hz a 3 kHz). Cada combinación consta de dos señales senoidales: una de un grupo bajo de frecuencias (697 Hz, 770 Hz, 852 Hz, 941 Hz) y otra de un grupo alto (1209 Hz, 1336 Hz, 1477 Hz, 1633 Hz). En la **figura 1** se muestra la matriz de

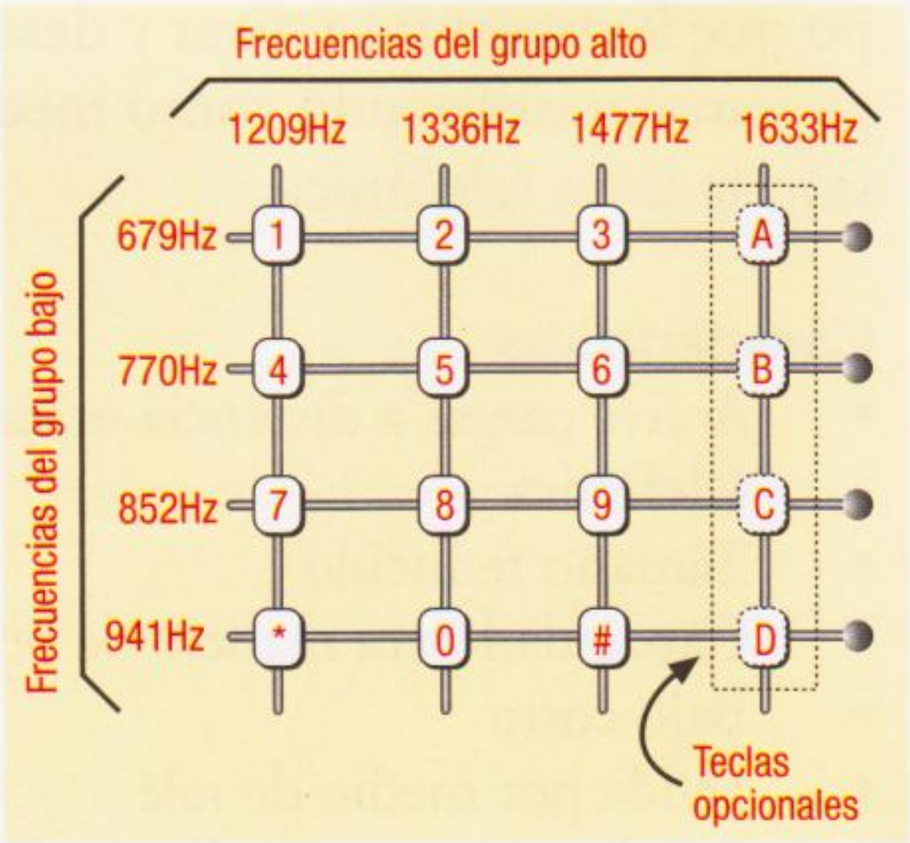


Figura 15.1 Frecuencias del sistema DTMF

frecuencias de un teclado marcador de tonos. Las teclas A, B, C y D se utilizan para tareas especiales y, normalmente, no forman parte de los teclados comunes.

Al pulsar el dígito «5» por ejemplo, se envían simultáneamente a través de la línea telefónica un tono bajo de 770 Hz y un tono alto de 1.336 Hz. Estos tonos son decodificados en la central telefónica para identificar el dígito marcado. Al pulsar dos o más teclas de una misma fila o columna, se genera un solo tono (el correspondiente a esa fila o columna). La pulsación de teclas diagonales no genera tonos.

La señalización DTMF tiene varias ventajas sobre la de pulsos, incluyendo una mayor rapidez de marcado y la posibilidad de enviar señales de control a través de la línea telefónica. La marcación de tonos se distingue fácilmente por los sonidos característicos que genera al pulsar cada entrada.

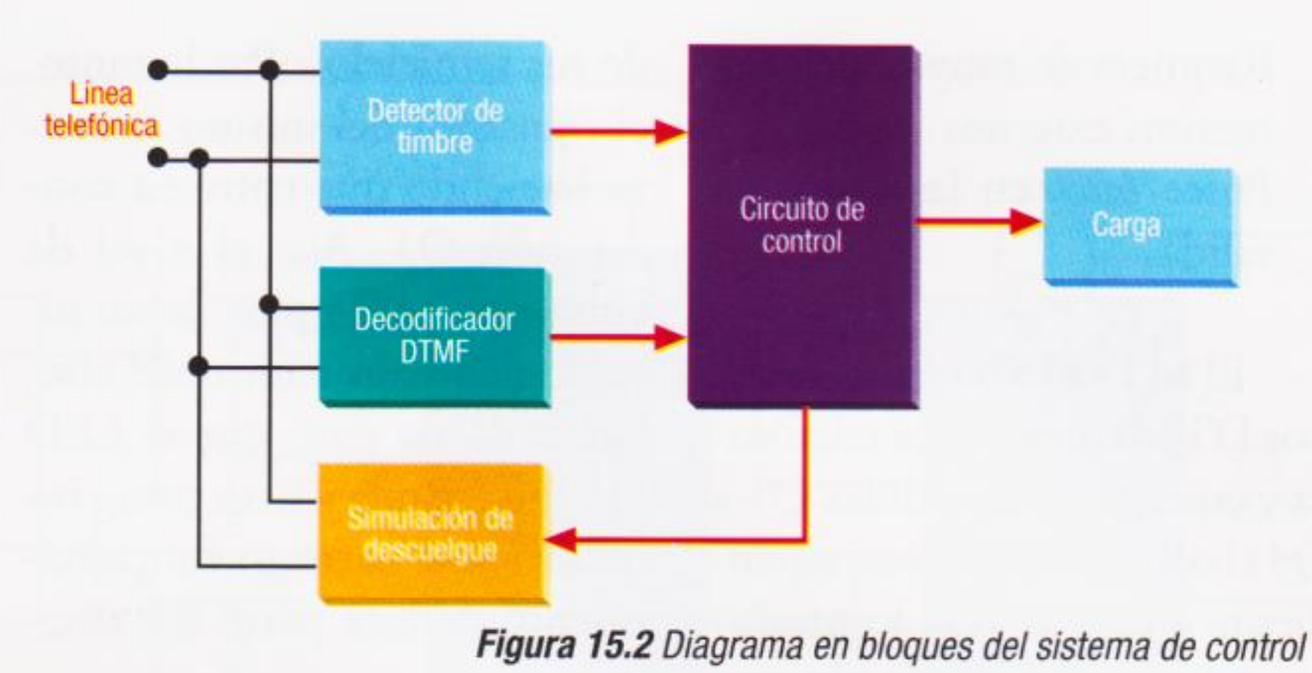


Figura 15.2 *Diagrama en bloques del sistema de control*

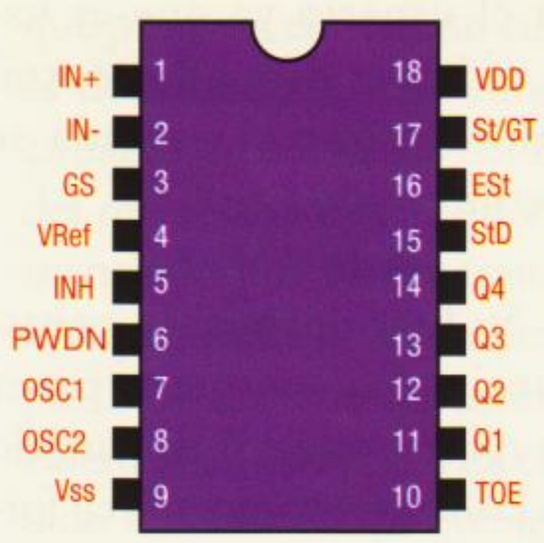
Descripción del circuito

El circuito debe estar permanentemente conectado a la línea para que pueda detectar el momento en que recibe su llamada. Por lo tanto, debe tener un sistema que le permita detectar la señal de timbre y por consiguiente, también debe tener un sistema que le permita simular que se ha descolgado el teléfono para que la central deje de enviar dicha señal. Como las órdenes de control se envían mediante tonos DTMF, también se debe incluir un sistema de decodificación de tonos que le permita al sistema de control tomar la decisión de si activa o no la carga de salida. En la **figura 15.2** se muestra el diagrama en bloques del sistema.

El decodificador de tonos MT88L70

Dado que la parte del circuito encargada de hacer la decodificación de los tonos es quizás la más compleja, se ha preferido utilizar un circuito integrado especial para dicha tarea. En este caso específico, utilice el decodificador MT88L70 de la empresa *Mitel*, el cual presenta las características necesarias para la tarea y además, se puede conseguir en el mercado fácilmente. En la **figura 15.3** se muestra su diagrama de pines correspondiente. Dentro de sus principales características se encuentran:

- Opera con fuente de alimentación entre 2,7 y 3,6 voltios
- Recepción de todos los tonos DTMF
- Bajo consumo de potencia



Descripción de pines		
1	IN+	**Non inverting input.** Entrada de señal no invertida
2	IN-	**Inverting input.** Entrada de señal invertida
3	GS	**Gain select.** Ajuste de ganancia
4	Vref	**Reference Voltage.** Es una salida con un voltaje igual a la mitad de la fuente
5	INH	**Inhibit.** Un lógico alto en este pin prohibe la detección de los tonos correspondientes a las teclas A, B, C y D.
6	PWDN	**Power Down.** Un lógico alto en este pin pone el dispositivo en modo de bajo consumo.
7	OSC1	**Clock.** Conexión del cristal de 3,579545 MHz
8	OSC2	**Clock**
9	Vss	**GND**
10	TOE	**Three State Output Enable.** Un lógico bajo en este pin pone las salidas en alta impedancia. Un lógico alto las habilita.
11-14	Q1-Q4	**Data output.** Salida de datos. Mantiene memorizado el último código recibido.
15	StD	**Delayed Steering.** Genera un pulso alto cuando recibe un tono válido y la salida actualiza el código recibido
16	ESt	**Early Steering.** Presenta un lógico alto cuando detecta un tono válido
17	St/GT	**Steering Input/Guard Time.** Ajusta el nivel de sensibilidad.
18	Vdd	**Power Supply.** Fuente positiva entre 2,7 y 3,6 voltios.

Figura 15.3 *Circuito decodificador DTMF MT88L70*

- Requiere de muy pocos elementos externos
- Posee *latch* en las líneas de salida

El MT88L70 recibe los tonos DTMF de la línea telefónica y entrega en sus salidas Q1 a Q4 el código binario correspondiente a la tecla que lo produce. Además, posee un pin que genera un pulso positivo cada vez que recibe un tono válido (StD). En la **figura 15.4** se muestra la tabla de códigos correspondiente que entrega este circuito integrado según el tono recibido y según la configuración de sus pines de control.

Funcionamiento

Cada uno de los bloques mencionados en la descripción del equipo posee un circuito equivalente que se encarga de hacer la tarea respectiva. En la **figura 15.5** se muestra el diagrama esquemático del equipo y vamos ahora a explicar cada una de sus partes.

Detector de la señal de timbre

La señal de timbre que envía la central telefónica, cuando se recibe una llamada, tiene aproximadamente 90 VCA y una frecuencia de 20 Hz. Para detectarla se utiliza el circuito conformado por el condensador C1, la resistencia R1, el optoacoplador IC1, el transistor Q1 y la resistencia R4.

La idea es que la señal alterna polariza correctamente el diodo del optoacoplador en uno de sus semiciclos. Por lo tanto, el transistor del mismo se activa haciendo que entre en conducción Q1. Así, el nivel de voltaje sobre R4 pasa de un nivel lógico bajo a un nivel alto, haciendo de paso que el LED D1 se encienda y sirva como indicador del correcto funcionamiento de esta parte del sistema.

Cuando la línea telefónica se encuentra en estado normal (colgado), el voltaje presente sobre la línea es de 48VCC y cuando se levanta el auricular (descolgado), dicho nivel cae por debajo de 10VCC. Por ello, el circuito de detección del timbre debe utilizar el condensador C1 para que la componente CC de la línea sea filtrada y no se produzca una sobrecarga de la misma.

Decodificador de tonos. Este circuito, como ya se mencionó, está construido alrededor de un circuito integrado MT88L70 (IC3), el cual utiliza unos pocos componentes externos para hacer su trabajo. Su conexión a la línea telefónica se hace a través del condensador C2 lo que evita problemas con el voltaje CC de la línea y además, deja pasar las frecuencias de los tonos DTMF.

Circuito de control. Está construido alrededor de un microcontrolador PIC16F84. Su principal función es recibir el código binario que entrega el decodificador de tonos y de acuerdo a los dígitos recibidos, decidir si activa o no la carga de salida. También es el encargado de verificar si llegó una señal de timbre en su pin RB7 para activar el circuito que simula que se ha descolgado la bocina del teléfono.

Cuando el PIC16F84 detecta que se recibió una señal de timbre válida, inicia una temporización de 10 segundos aproximadamente, tiempo durante el cual permanece activado el circuito que simula que el teléfono ha sido descolgado.

Este tiempo es empleado por el usuario que ha hecho la llamada para enviar los tonos DTMF que harán que la carga sea activada o desactivada. El hecho de tener un microcontrolador en el circuito hace que las posibilidades de control sean muy grandes; por ejemplo, usted puede tener una clave de acceso para que sea admitida una orden de encender o apagar la carga.

Simulación de teléfono descolgado. Este sistema permite establecer la comunicación con el usuario ya que es usted quien le dice a la central telefónica que la llamada se ha contestado.

Su funcionamiento es bastante simple y consiste en poner una resistencia (R3) en paralelo con la línea telefónica, de tal forma que la central vea una impedancia equivalente similar a la

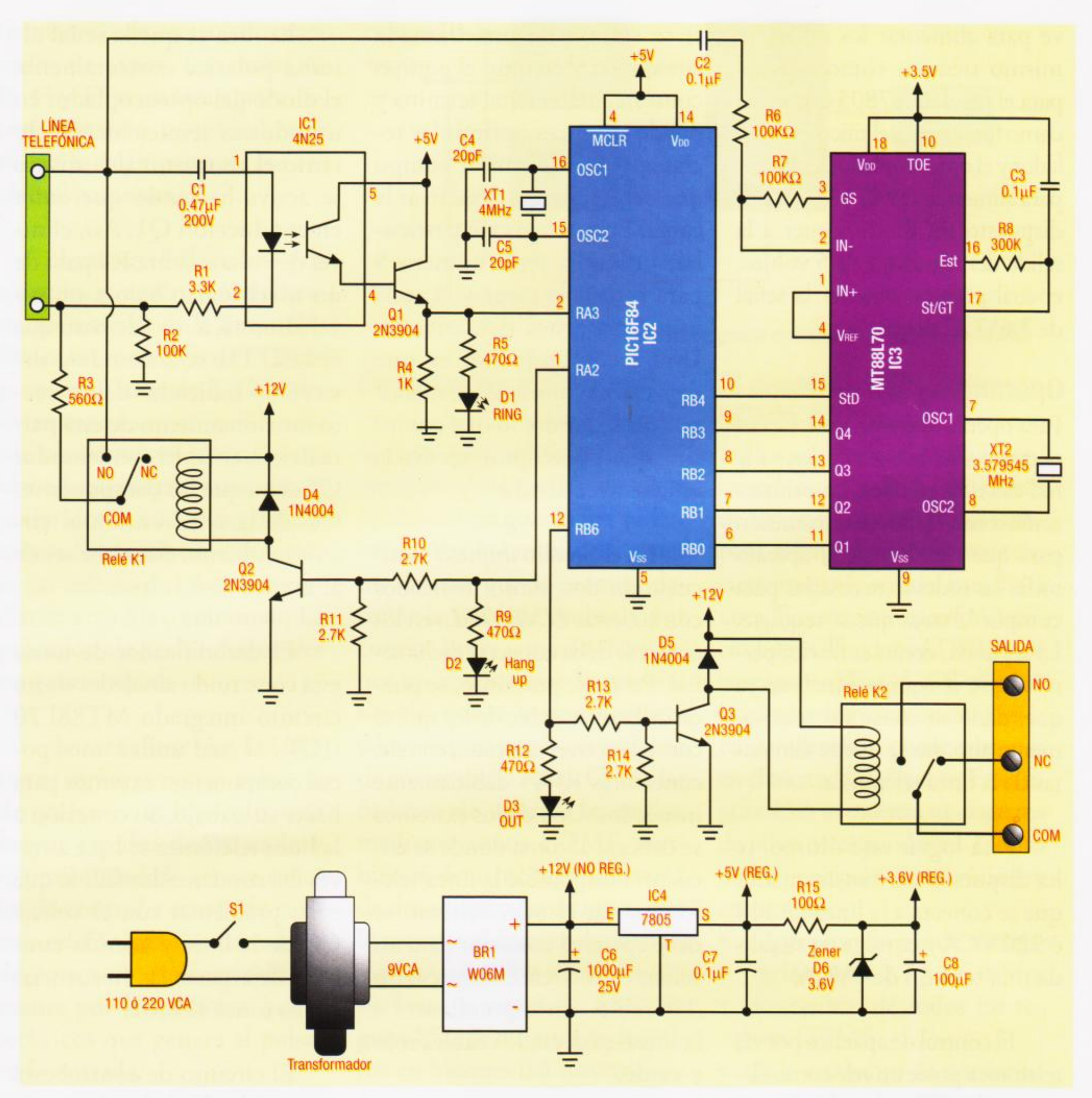


Figura 15.5 *Diagrama esquemático del circuito*

que presenta el teléfono cuando es descolgado. Dicha conexión se hace a través de los contactos del relé K1, el cual, a su vez, es manejado por el del transistor Q2 que recibe la orden desde el pin RB6 del microcontrolador. Además, en este pin se ha conectado el LED D2 lo que le permite establecer si el equipo ha contestado la llamada.

Conexión de la carga. Cuando el circuito de control recibe una orden válida para encender o apagar la carga de salida, solamente debe activar o desactivar el relé K2 que es el encargado de hacer la conmutación de la misma. La utilización del relé permite que el circuito maneje una carga de cualquier tipo, independientemente de cual sea su naturaleza. Para confirmar el estado del relé, se ha conectado el LED D3 en el mismo pin que lo controla.

Fuente de alimentación. El sistema se puede alimentar de la red de 110 ó 220 VCA mediante un transformador que entregue en su salida 9 VCA. Una vez rectificada dicha señal, se obtiene un voltaje cercano a los 12VCC que sir-

ve para alimentar los relés y al mismo tiempo, como entrada para el regulador 7805 que actúa como fuente para el microcontrolador y el optoacoplador. Además, para alimentar el MT88L70 se ha dispuesto un diodo Zener a la salida del regulador de 5 voltios, el cual permite obtener la señal de 3,6VCC necesaria.

Operación

Para operar este equipo debe conectarse a la línea telefónica y a la red eléctrica pública. La primera actúa como medio de transmisión para que usted envíe al aparato todas las órdenes necesarias para cumplir la tarea que se requiere. La segunda, como es obvio, permite que el equipo funcione ya que, dado su consumo relativamente alto, no se puede alimentar de la línea telefónica.

Para lograr esto último, se ha dispuesto un transformador que se conecta a la línea de 110 ó 220 VCA y entrega en su salida una tensión de 9VCA.

El control de aparatos por vía telefónica posee un relé como elemento de salida, el cual le permite activar y desactivar cualquier tipo de carga conectada a través de sus contactos. La idea es que usted pueda, desde algún sitio lejano, marcar el número de teléfono en que está conectado el equipo y cuando él conteste, poder dar las órdenes de encender o apagar la carga.

Cuando usted hace la llamada, escucha el timbre como si se tratara de una llamada cualquiera y cuando el equipo contesta, dicha señal termina y puede entonces oprimir las teclas que le indican al equipo que debe activar o desactivar la carga. En nuestro caso particular, utilice la tecla número 5 para activar la carga y la tecla número 6 para desactivarla. Una recomendación es que deje la tecla oprimida al menos por un segundo, lo que garantiza que el receptor reciba la señal correctamente.

En el circuito impreso se encuentran dos puntos marcados con las letras *LINEATEL.*, en los cuales se debe conectar dicha señal. Para este propósito, se puede utilizar un cable de los que se consiguen en el mercado, con sus conectores RJ-45 debidamente instalados. Uno de los extremos se conecta al toma donde se encuentra disponible la línea telefónica. En el otro extremo se debe cortar el conector para así soldar en el circuito impreso los dos cables correspondientes a la línea, es decir, los cables rojo y verde.

A través de dichos terminales llegan al circuito la señal de timbre y los tonos DTMF.

Para detectar la señal de timbre, que tiene aproximadamente 90 VCA y una frecuencia de 20 Hz, se utiliza el circuito conformado por el condensador C1, la resistencia R1, el optoacoplador IC1, el transistor Q1 y la resistencia R4.

La idea es que la señal alterna polarice correctamente el diodo del optoacoplador en uno de sus semiciclos. Por lo tanto el transistor del mismo se activa haciendo que entre en conducción Q1. Así, el nivel de voltaje sobre R4 pasa de un nivel lógico bajo a un nivel alto, haciendo de paso que el LED D1 se encienda y sirva como indicador del correcto funcionamiento de esta parte del sistema. El condensador C1 evita que las componentes CC de la señal entren al circuito. Además, éste debe ser de al menos 200 voltios.

El decodificador de tonos está construido alrededor de un circuito integrado MT88L70 (IC3), el cual utiliza unos pocos componentes externos para hacer su trabajo. Su conexión a la línea telefónica se hace a través del condensador C2, lo que evita problemas con el voltaje CC de la línea y además también deja pasar las frecuencias de los tonos DTMF.

El circuito de control está construido alrededor de un microcontrolador PIC16F84. Su principal función es recibir el código binario que entrega el decodificador de tonos y de acuerdo al dígito recibido, decidir si activa o no la carga de salida. También es el encargado de verificar si llegó una señal de timbre en su pin RA3 para activar el circuito que simula que se ha descolgado el auricular del teléfono.

Dicha operación también inicia una temporización de aproximadamente 10 segundos, tiempo durante el cual el usuario debe oprimir la tecla que indica la operación que desea llevar a cabo. El uso de un microcontrolador también abre un mundo de posibilidades bastante amplio; por ejemplo, se puede escribir un programa que obligue al usuario a entrar una clave de seguridad antes de poder cambiar el estado de la carga que está siendo manejada por el equipo.

Cuando el equipo recibe una llamada, el microcontrolador activa el relé, el cual se encarga de conectar la resistencia R3 en paralelo con la línea telefónica, lo que le indica a la central que se ha contestado. Esto es posible gracias a que dicha resistencia simula una impedancia similar a la que presenta un teléfono convencional cuando se levanta el auricular.

Para la conexión y desconexión de la carga se utiliza simplemente un relé de 12 voltios, el cual es accionado desde el microcontrolador mediante el transistor Q3. Esto permite que usted pueda manejar cualquier tipo de carga. Por otra parte, el sistema se debe alimentar con 9VCA, los cuales se obtienen del secundario del transformador.

Otra opción, consiste en utilizar una fuente de 12VCC que se tenga disponible en el sitio de trabajo. En la **figura 15.6** se muestra la forma en que se debe conectar el equipo para obtener un correcto funcionamiento.

Para la prueba del equipo se requieren dos líneas telefónicas, una para conectar el equipo de control y otra para que el usuario haga la llamada y verifique que el sistema funciona correctamente. Obviamente, su teléfono debe ser de marcado por tonos y la línea donde se conecta el equipo de control debe ser una línea convencional. En ningún caso se puede conectar a una línea derivada de un PBX.

Descripción del programa

Dado que el funcionamiento del equipo es bastante simple, el programa también lo es. Básicamente, el microcontrolador debe estar pendiente de que aparezca una señal de nivel alto en el pin RA3 lo que indica que se recibe una llamada.

Posteriormente, el aparato debe activar el relé que se maneja desde el pin RA2 y mantenerlo así durante aproximadamente 10 segundos, tiempo durante el cual se debe revisar el pin RB4 que es donde se debe registrar un pulso de nivel alto proveniente del integrado MT88L70, y que indica que se recibió un código DTMF válido.

Luego de lo anterior, el microcontrolador sólo debe leer el dato que está presente en sus pines RB3 a RB0, el cual corresponde al dígito que se ha recibido. Esta tarea es bastante sencilla ya que el MT88L70 posee cerrojos (*latches*) en sus salidas. Por último, el microcontrolador debe revisar si el código corresponde al número 5, para activar el relé de salida, ó al número 6, que indica que se debe desactivar. Una característica especial es que utilizamos la memoria EEPROM de datos del PIC16F84 para almacenar el estado que debe tener la carga en un momento determinado.

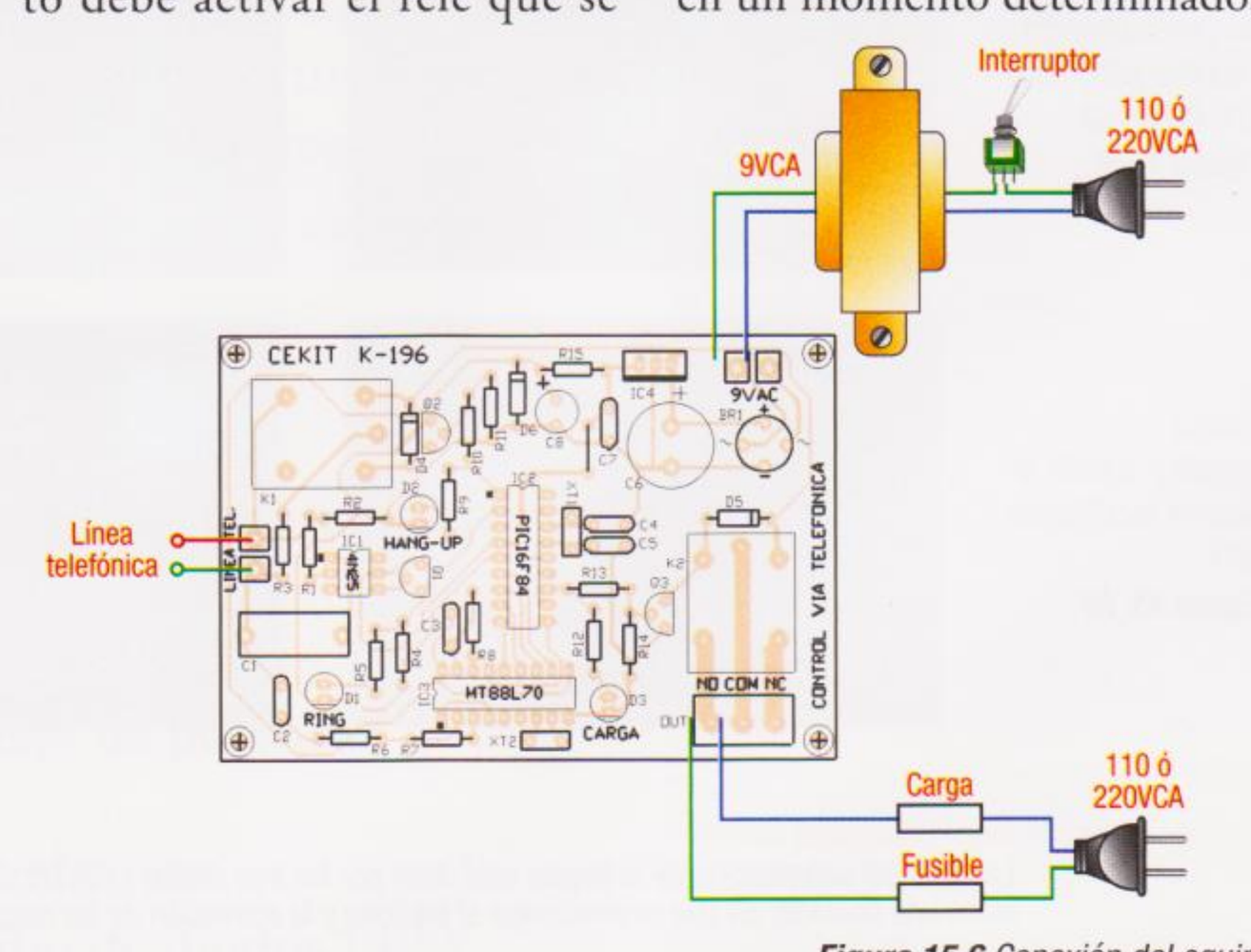


Figura 15.6 *Conexión del equipo*

Ensamblaje del proyecto

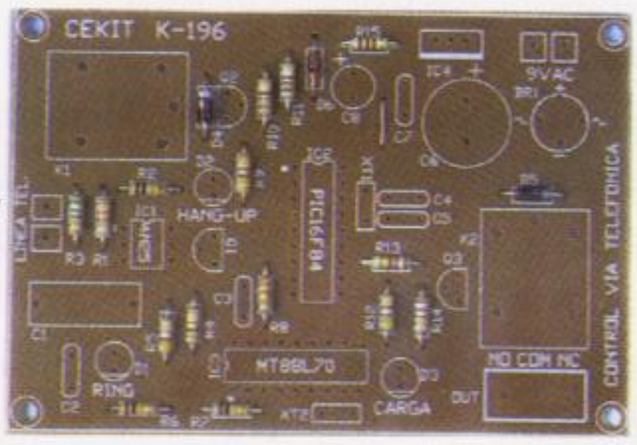
Figura 15.7
Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el sistema de control de aparatos por vía telefónica.

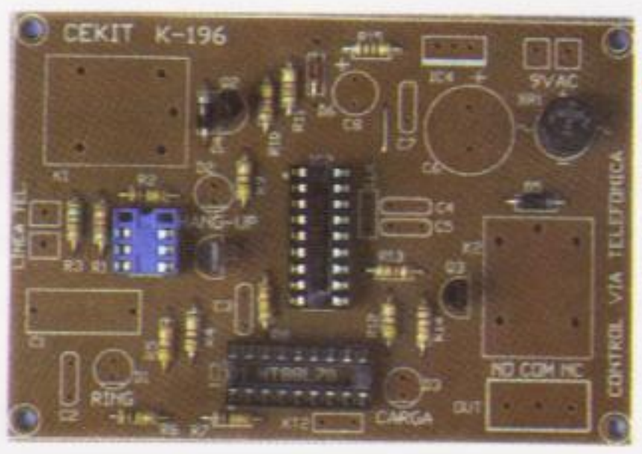
Por medio de este circuito se podrá controlar el encendido de dispositivos a distancia, tan solo con una llamada telefónica. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

Ref.	Descripción	Cant.
IC1	Optoacoplador 4N25	1
IC2	Microcontolador PIC16F84	1
IC3	Circuito integrado MT88L70	1
IC4	Regulador fijo de +5V, 7805	1
Q1 a Q3	Transistores NPN, 2N3904	3
BR1	Puente rectificador de 1A, W06M	1
D1 a D3	Diodos LED rojos de 5mm	3
D4 y D5	Diodos rectificadores 1N4004	2
D6	Diodo zener de 3,6V a 1/2W	1
C1	Condensador cerámico de 0,47µF/250V	1
C2,C3 y C7	Condensadores cerámicos de 0,1µF/50V	3
C4 y C5	Condensadores cerámicos de 20pF/50V	2
C6	Condensador electrolítico de 1000µF/25V	1
C8	Condensador electrolítico de 100µF/25V	1
R1	Resistencia de 3.3k-1/4W	1
R2,R6 y R7	Resistencias de 100k-1/4W	3
R3	Resistencia de 560-1/4W	1
R4	Resistencia de 1k-1/4W	1
R5,R9 y R12	Resistencias de 470-1/4W	3
R8	Resistencia de 300k-1/4W	1
R10,R11,R13 y R14	Resistencia de 2.7k-1/4W	4
R15	Resistencia de 100-1/4W	1
XT1	Cristal de cuarzo de 4 MHz	1
XT2	Cristal de cuarzo de 3,579545 MHz	1
K1 y K2	Relés de 12V	2
K-196	Circuito impreso CEKIT	1
	Interruptor de codillo de 2 posiciones	1
	Conector de tornillo de 3 pines	1
	Base para integrado de 8 pines	1
	Base para integrado de 18 pines	2
	Transformador con Primario: 110 ó 220 VCA Secundario: 9VCA	1
	Cable de potencia con enchufe	1
	Terminal para circuito impreso (espadines)	4

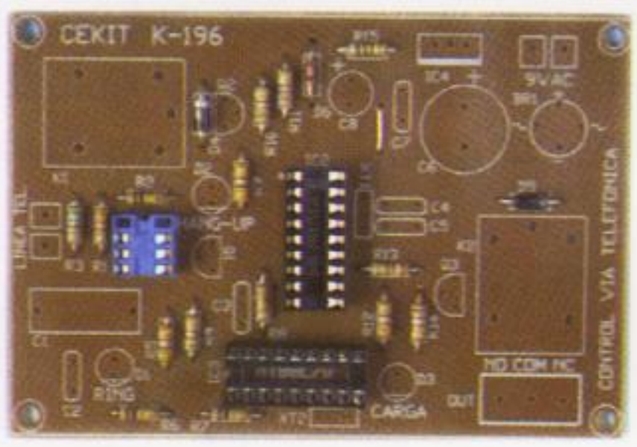
Paso 1
Instale y suelde los puentes de alambre, las quince resistencias R1 a R15, y los diodos zener y rectificadores D4 a D6.
Figura 15.8

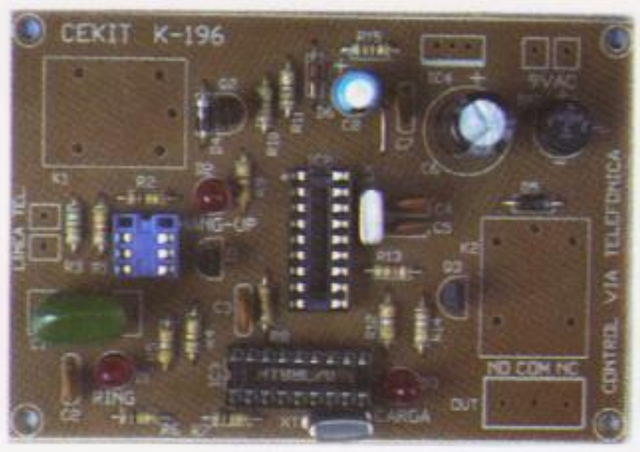




Paso 4
Instale y suelde los transistores Q1, Q2 y Q3.
Figura 15.11

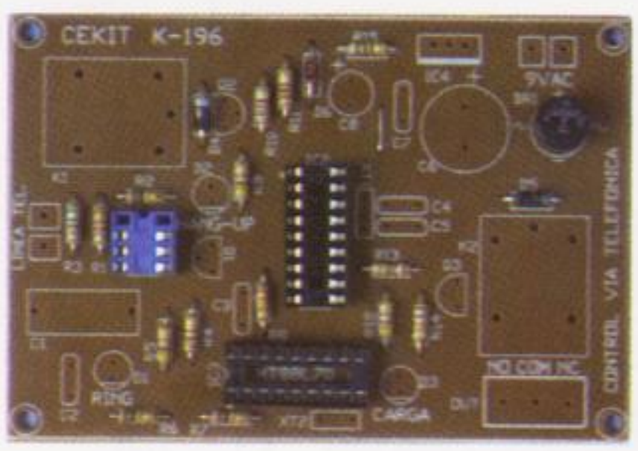
Paso 2
Luego instale y suelde las bases para los circuitos integrados IC1, IC2 e IC3.
Figura 15.9





Paso 5
Fije y suelde los diodos LED, los condensadores C1 a C5 y C7, los cristales de cuarzo XT1 y XT2 y los condensadores electrolíticos C6 y C8.
Figura 15.12

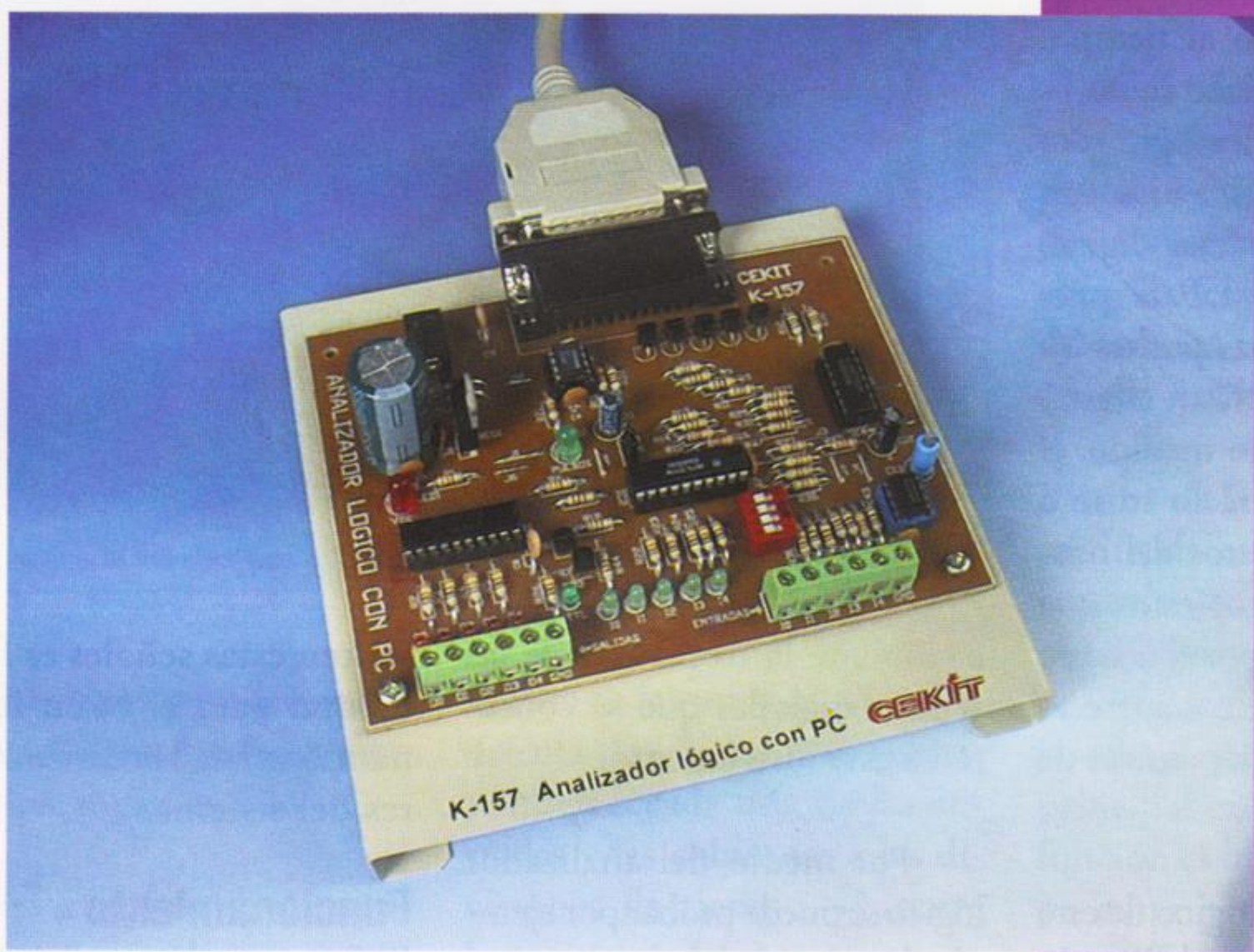
Paso 3
Instale y suelde el puente rectificador BR1.
Figura 15.10



Paso 6
Aspecto del proyecto luego de instalar y soldar los relés, el regulador IC4, el conector de salida y los espadines.
Figura 15.13

Paso 7. Pruebas finales
La principal señalización de la tarjeta está dada por los tres diodos LED: D1, D2 y D3. Ellos indican el momento en que entra una llamada, en que se descuelga el teléfono y la activación de las cargas.

PROYECTO 16



ANALIZADOR LÓGICO CON PC

Ensamble esta herramienta de gran utilidad para su laboratorio. Por medio de este analizador lógico, que se conecta al puerto paralelo de su computadora, puede evaluar el funcionamiento de los circuitos digitales, conocer su frecuencia, su ciclo útil, o simplemente compararlo con otras señales dentro del mismo circuito de prueba.

Características

- Útil aparato para su laboratorio de electrónica digital y microcontroladores
- Graficación de las lecturas en una computadora
- Lectura visual de los retardos entre señales
- Lectura de frecuencia
- Alta velocidad

Una analizador lógico es un instrumento de medida que sirve para la visualización de varias señales digitales en una pantalla, de tal modo que se puedan confrontar unas con otras y analizarlas con respecto al tiempo. Esta herramienta tiene cierta similitud con un osciloscopio, sólo que las señales utilizadas son únicamente de carácter digital, es decir, unos o ceros. Para quienes han utilizado las puntas lógicas, las cuales indican el estado lógico del punto medido, el analizador mostrará lo mismo pero en varios puntos del mismo circuito, puntos estos que son seleccionados por el usuario y cuyos estados lógicos aparecerán dibujados en la pantalla de la computadora.

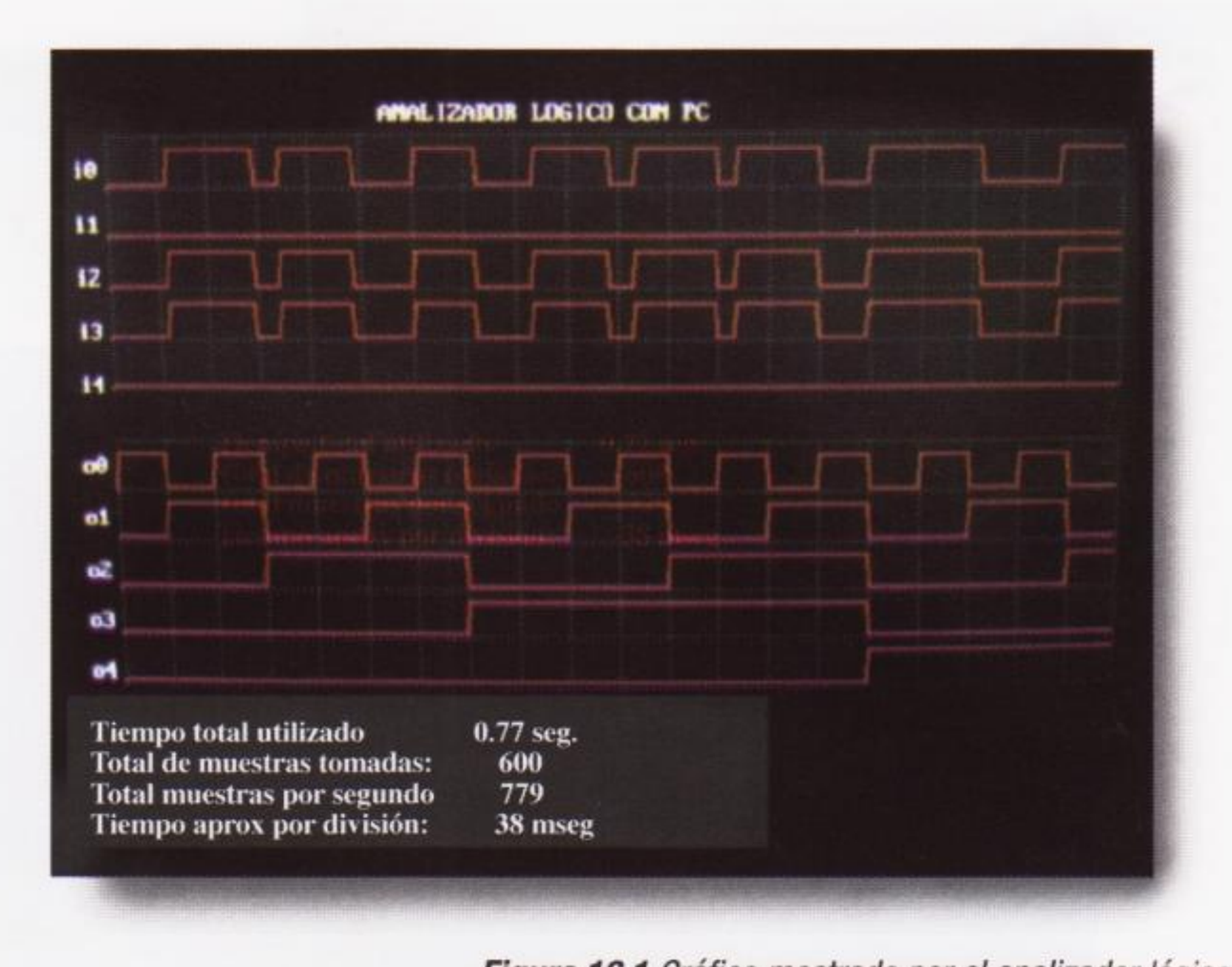


Figura 16.1 *Gráfico mostrado por el analizador lógico*

El analizador lógico de este proyecto puede ser utilizado como herramienta de laboratorio para probar circuitos digitales. Lo que podía leerse con una punta lógica, ahora puede hacerse con esta herramienta con la salvedad de que no se obtiene el estado de un solo punto del circuito sino de cinco a la vez,; además, podemos analizar dichos estados lógicos con respecto al tiempo suministrado por la computadora, **figura 16.1**.

Otra de las propiedades del analizador lógico consiste en la posibilidad de cálculo de las frecuencias de los datos leídos, tal como un frecuencímetro. Con los tiempos suministrados por el programa, puede calcularse fácilmente la frecuencia de las señales digitales que se conecten a este instrumento.

Por medio del analizador lógico se puede probar por ejemplo, la velocidad de los circuitos integrados, los cuales presentan un retardo al cambio de señal dependiendo de la calidad de los mismos. En la figura 16.2 se ilustra la diferencia de respuesta entre dos circuitos integrados diferentes.

En los circuitos con microcontroladores este instrumento se puede utilizar para visualizar los tiempos, frecuencias y desfases exactos, de las distintas señales digitales que intervienen dentro de un sistema. En un circuito digital la exactitud con que se presenten estas señales es fundamental para el buen funcionamiento de las distintas partes del sistema.

Funcionamiento

El funcionamiento del analizador lógico con PC es muy sencillo. Observe en la **figura 16.3** el diagrama de bloques que describe la operación del circuito. Es suficiente conectar las entradas del analizador a los

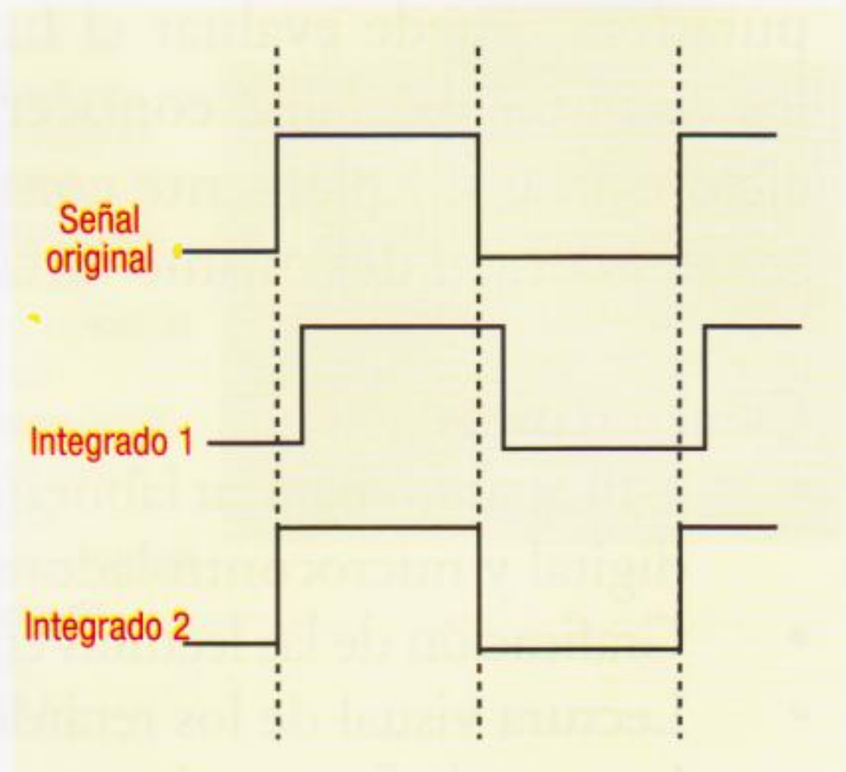


Figura 16.2 *Medida del retardo de operación de circuitos integrados*

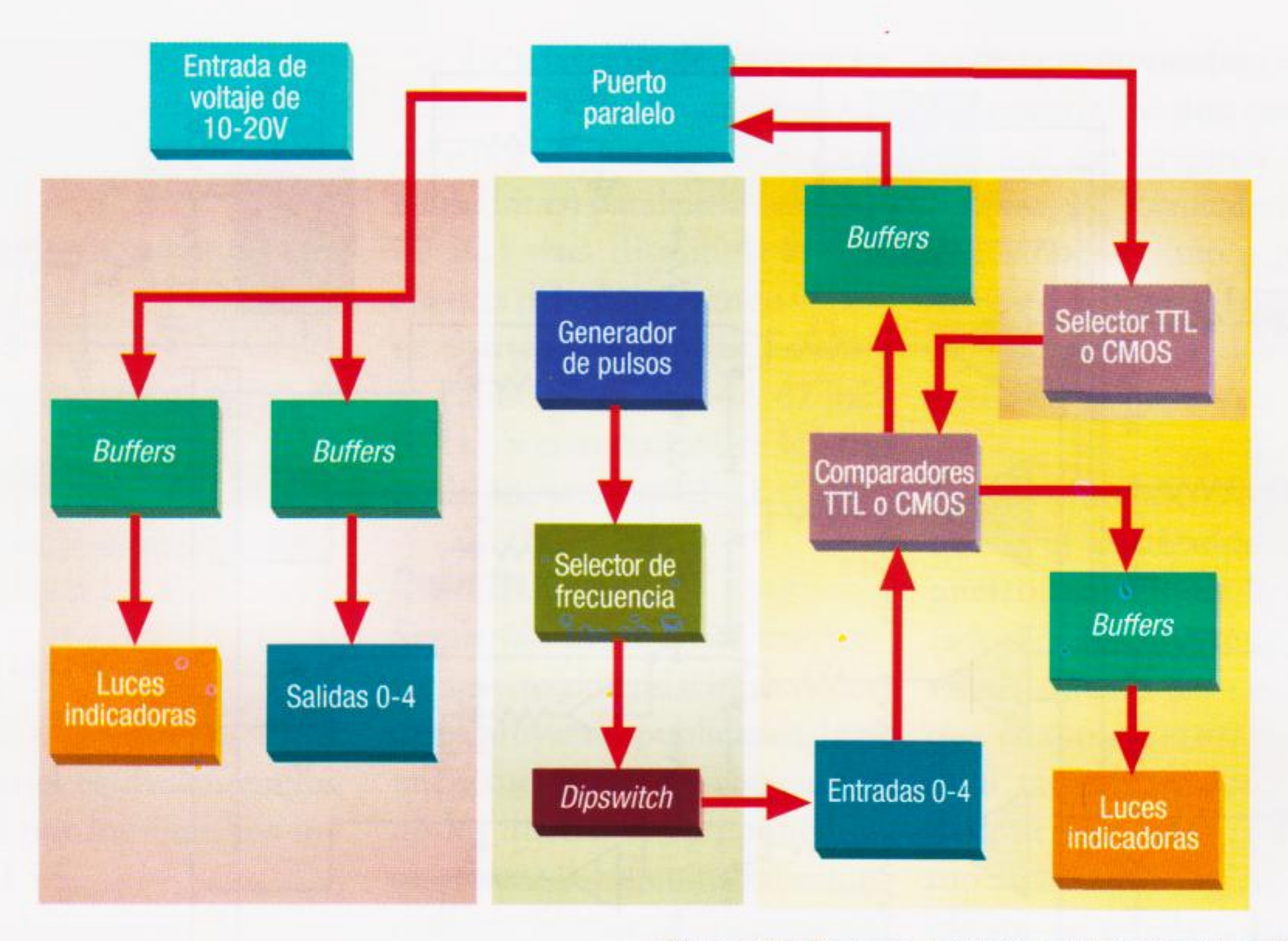


Figura 16.3 *Diagrama de bloques del circuito del analizador lógico*

puntos del circuito que deseamos probar y hacer la lectura por medio del programa cargado en la computadora. Así entonces, con el gráfico mostrado, podemos analizar las señales con respecto al tiempo, a la frecuencia, o sencillamente compararlas con las demás.

Con los datos suministrados por el analizador, es posible hacer un diagnóstico y detectar en qué puntos está fallando el circuito, o simplemente, cuáles señales no están de acuerdo a lo deseado.

Adicionalmente, el analizador lógico de este proyecto cuenta con una serie de salidas que pueden ser utilizadas para excitar el circuito digital que se va a probar. Estas salidas entregan estados lógicos correspondientes a un contador binario de 0 a 20, logrando con esto una buena combinación de unos y ceros que sirven para probar cualquier circuito digital que permita la entrada de estados lógicos.

Descripción del circuito

En la **figura 16.4** aparece el diagrama esquemático del circuito del analizador lógico con PC. Como se puede apreciar, el proyecto está compuesto básicamente por cinco circuitos integrados entre los que se destacan IC1 e IC2 que son compuertas YES como *buffer*, cuya función es la de dar potencia a las señales digitales de entrada y salida así como la de suministrar la corriente suficiente para encender los LED indicadores.

De otra parte, se cuenta con dos circuitos integrados, IC4 e IC5, que contienen cinco amplificadores operacionales en total, cuya principal característica es su alta velocidad de operación, permitiendo leer frecuencias altas en las entradas del analizador. Son utilizados como comparadores para poder seleccionar entre la tecnología TTL y la CMOS, cuyos niveles de voltaje para diferenciar los niveles lógicos, es decir los unos y ceros, son diferentes.

Para seleccionar una tecnología o la otra, se cambian las referencias de voltaje de cada uno de los comparadores; esto se hace por medio del transistor TR7 a través de un pulso que es enviado desde la misma computadora con una de las opciones del programa. Cuando este transistor entra en operación, conecta la resistencia R44 del voltaje de referencia haciendo que este nivel se dismi-

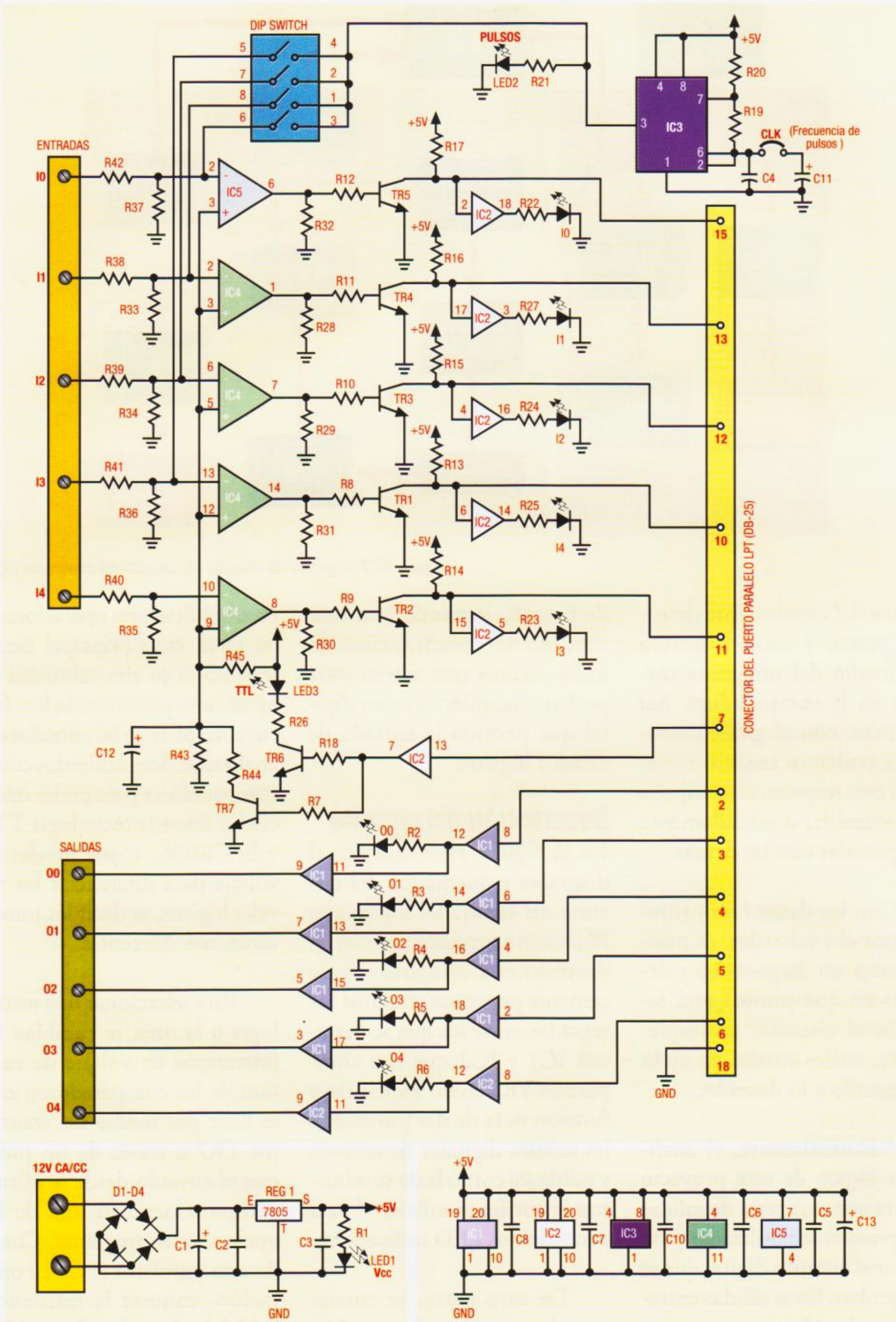


Figura 16.4 Diagrama esquemático del analizador lógico

Pines DB25 macho	Pines DB25 macho	Ent. / Sal.	Bit
2	2	Sal.	0
3	3	Sal.	1
4	4	Sal.	2
5	5	Sal.	3
6	6	Sal.	4
7	7	Sal.	5
10	10	Ent.	6
11	11	Ent.	7
12	12	Ent.	5
13	13	Ent.	4
15	15	Ent.	3
18	18	GND	

Figura 16.5 *Elaboración del cable de comunicación*

nuya hasta el equivalente a los utilizados por los circuitos integrados TTL.

El circuito también posee un generador de onda cuadrada de dos frecuencias distintas, con el único objeto de ser utilizado como prueba de funcionamiento del analizador lógico. Para probarlo se cuenta con cuatro interruptores miniatura (*dipswitch*) donde cada uno de ellos conecta el generador a una de las primeras cuatro entradas del analizador, y así, poder visualizar dicha onda en la pantalla de la computadora.

Con la frecuencia baja, la señal emitida por el generador se puede ver fácilmente en la pantalla haciendo lecturas con retardo de tiempo (esto fue probado en un Pentium de 166 MHz y difiere en otras computadoras), además pueden verse los destellos a simple vista. Con la frecuencia alta deberán efectuarse lecturas de alta velocidad en la computadora; esto se logra con una de las opciones del programa.

La selección de frecuencia se hace por medio de un *jumper* o puente que agrega un condensador en paralelo al integrado IC3 para disminuir la frecuencia del reloj. Cuando se tiene el *jumper* en la posición alta, la capacitancia es de 47 uF (C11) y cuando está en la otra posición, será de 0.1 uF (C4).

Construcción

Siguiendo las mismas instrucciones y recomendaciones de los otros proyectos publicados, instale y suelde con mucho cuidado los diferentes componentes que forman el circuito, tal como muestra la guía de montaje de la página 142, respetando las polaridades, tanto de los condensadores como de los diodos y los LED indicadores, al igual que las posiciones indicadas.

Cable de comunicación. El cable utilizado para llevar y traer las señales hacia y desde el puerto de la computadora se elabora a partir de dos conectores machos de 25 pines (DB25) machos. La forma de hacer las conexiones es uno a uno; observe en la **figura 16.5** la relación de pines que deben alambrarse entre dichos conectores.

Pruebas iniciales. Antes de conectar el analizador lógico al puerto paralelo de la computadora, haga una revisión previa del estado del circuito, ya que cualquier error podría ser fatal tanto para los circuitos del puerto como para el mismo analizador lógico. Inicialmente se pueden probar las entradas haciendo uso del generador de pulsos del mismo circuito. Al conectarse a cada una de las entradas por medio de los interruptores miniatura, las luces indicadoras correspondientes a las entradas deberán parpadear a la misma frecuencia de dicho generador.

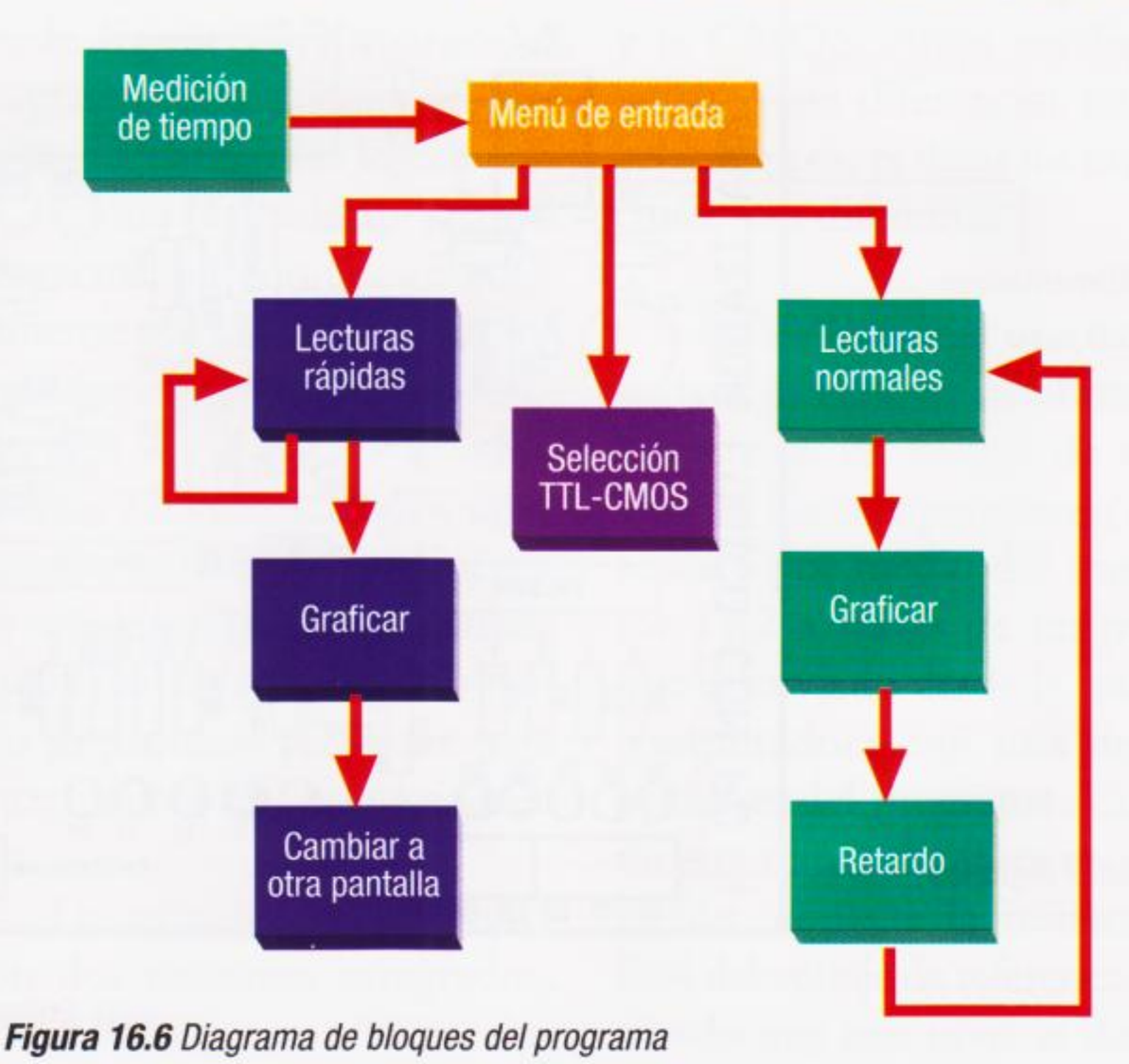


Figura 16.6 *Diagrama de bloques del programa*

Programa

El programa debe elaborarse en ambiente DOS ya que éste no comparte tiempos con otros programas como sucede en Windows. Si se hace bajo ambiente Windows, se tendría el problema de que los tiempos leídos no son reales porque este sistema operativo hace una secuencia de trabajo con cada uno de los programas que están funcionando en ese momento, inclusive con la misma pantalla de entrada a Windows. En este caso, el programa fue escrito en lenguaje C para DOS.

En la **figura 16.6** puede verse un diagrama de bloques que describe la relación entre las secciones del programa hecho. Observe que principalmente se tienen dos secuencias: la de lectura rápida y la de lectura con retardo.

Pruebas finales de la tarjeta

Una vez se ha probado el circuito electrónico del analizador, tal como se mencionó anteriormente, puede conectarse al puerto paralelo de la computadora (estando ambos apagados), por medio del cable de comunicación. Posteriormente, conecte la alimentación al analizador y encienda la computadora ejecutando el programa. Con esto, ya está listo el instrumento para empezar a probar y a analizar circuitos digitales.

En la **figura 16.7** puede observar la guía de montaje que puede utilizar para conectar los distintos elementos que componen el circuito y la ubicación de los conectores externos.

Las rutinas que componen este proyecto están disponibles en nuestra página de la internet, entre ellas se encuentran la de cálculo de tiempos, las de lecturas normal y rápida y el listado del encabezado general del programa. Veamos a continuación una breve descripción de las funciones y algunos listados de las mismas.

Declaración de librerías, funciones y variables. Este es el encabezado del programa y define los recursos, funciones y variables que serán utilizados durante la compilación del mismo, **figura 16.8**.

Rutina de cálculo de tiempo. Debido a que el analizador lógico puede instalarse en diferentes computadoras y que lo más seguro es que éstas difieran también de la velocidad de proceso, el programa ejecuta una prueba de tiempo, en la cual se hacen una

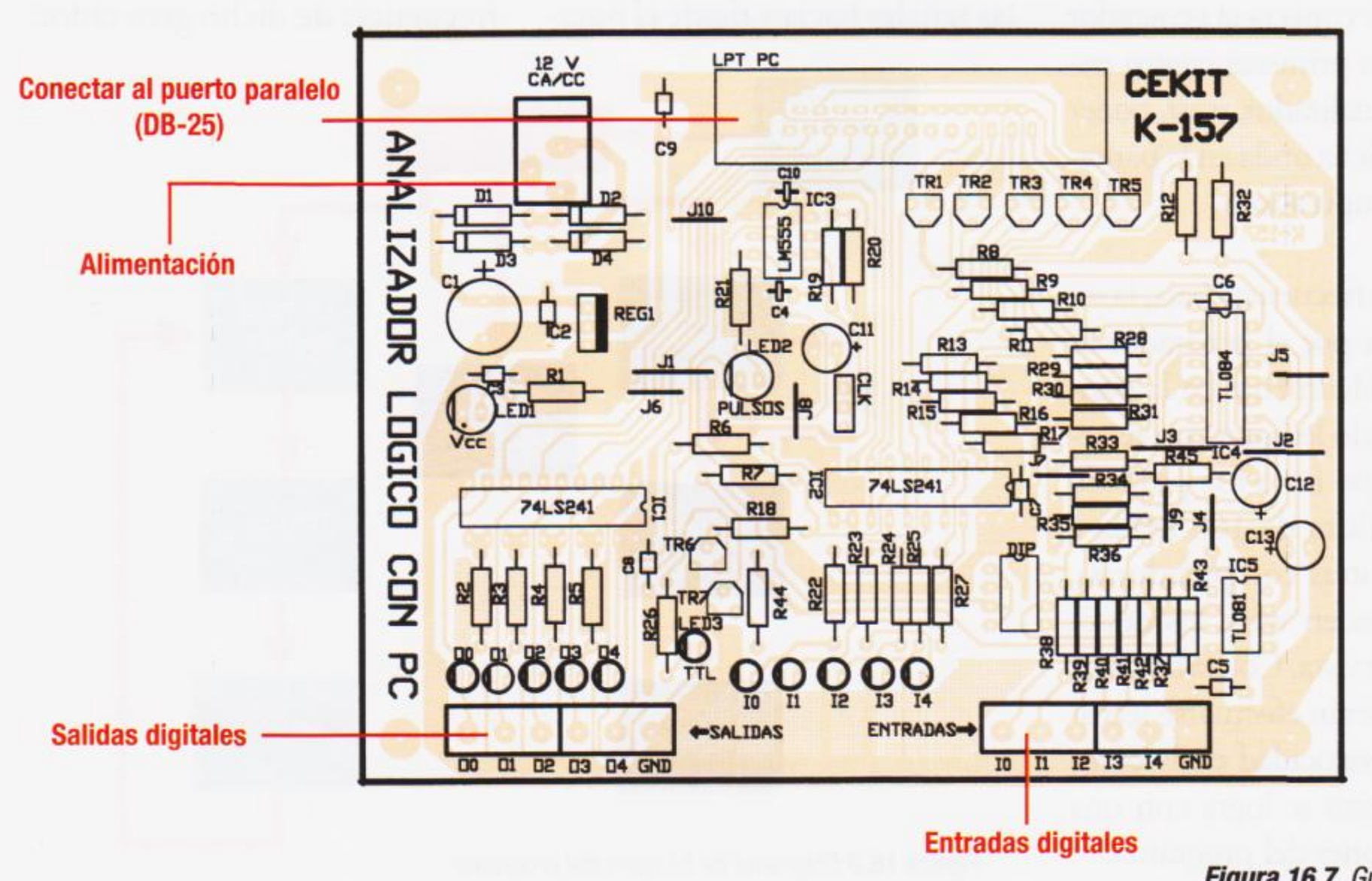


Figura 16.7 Guía de montaje

```
//Definición de librerías
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#include <graphics.h>
#include <stdlib.h>


//Declaración de funciones
void tiempo(void);          //Realiza cuentas de tiempo
void cuadricula(void);      //Dibuja cuadrícula en la pantalla
int modograf(void);         // Inicializa modo gráfico
void leerescribir();        // Lee y escribe valores en el puerto
void lecturarapida();       //hace lectura superrápida

//Declaración de variables
int h,i,j,k;
int retardo=1, TTL=32;
int valor[25000];
float tiempototal,tiempodelect;
double lecturas;
char texto[20];
struct  time t1,t2;
int gdriver = DETECT, gmode, error;
int ent,sal;
int in[5],out[5],y[10];

const P1=0x378,P2=0x379;   //Dirección de puertos LPT1
```

Figura 16.8 *Encabezado del programa*

serie de lecturas para detectar el tiempo utilizado por la computadora que se esté usando.

Por ejemplo, si la computadora está basada en un microprocesador 386, seguramente tardará más tiempo en hacer la misma cantidad de lecturas que otra basada en microprocesador Pentium.

Modo gráfico. Esta rutina es indispensable para poder utilizar la pantalla de la computadora en modo gráfico. Los cálculos de coordenadas están basados en una pantalla que permite una resolución de 640 x 480 pixels, resolución ésta muy común en monitores VGA y SVGA.

Divisiones de tiempo. Con el fin de poder medir el tiempo y hacer las comparaciones respectivas sobre las señales, el programa dibuja antes de la lectura de las mismas, una cuadrícula donde las filas corresponden a cada una de las entradas y las columnas a las divisiones de tiempo. Durante los cálculos de tiempo, se involucran la cantidad de lecturas y de puntos en los que se divide la pantalla.

Función de lectura normal. Esta función permite que la computadora haga lecturas con intervalos de tiempo que el usuario puede cambiar por medio de los caracteres '+' y '-'.

Esta rutina hace una lectura y la muestra directamente en la gráfica de la pantalla de la computadora antes de proceder a la siguiente lectura.

De cualquier manera, esta opción es útil cuando la frecuencia de las señales de entrada es baja comparada con la que puede leer la computadora que se esté utilizando. Una ventaja es poder graficar instantáneamente de acuerdo al valor leído en el analizador, cosa que no sucede con la opción de lectura rápida.

Cuando finalizan las 600 lecturas correspondientes a una pantalla, el programa muestra el tiempo total utilizado y la equivalencia en milisegundos de cada una de las 20 divisiones del gráfico.

Rutina de lectura rápida. Por medio de esta función se hacen lecturas rápidas del analizador lógico, permitiendo así, examinar circuitos con frecuencias digitales altas. La principal característica de esta función es que no utiliza tiempo extra para graficar las lecturas. El proceso se hace en dos etapas: lecturas consecutivas y graficación.

Primero se hacen 24.000 lecturas del estado lógico de los puntos que se van a medir en el circuito de prueba y son llevadas a posiciones de memoria, proceso que es mucho más rápido que el de graficar los resultados a la vez que son leídos. La otra etapa muestra los resultados actualmente almacenados en la memoria por medio de 40 pantallas, donde la primera corresponde a las primeras 600 lecturas del analizador, la segunda pantalla a las lecturas 601 a 1.200 y así sucesivamente hasta mostrar las 24.000 lecturas tomadas.

Ensamblaje del proyecto

Figura 16.9 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el analizador lógico con PC.*

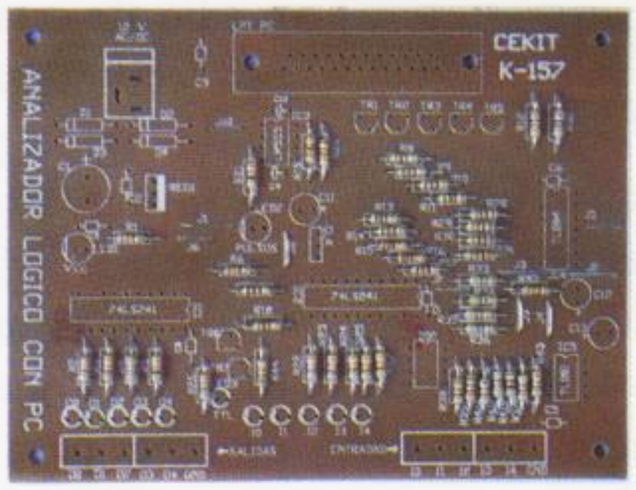
El reconocimiento y la lectura de varias señales de voltaje con niveles lógicos es de gran utilidad en el diseño y desarrollo de dispositivos con base en microcontroladores, y puede hacerse independientemente o de manera simultánea para varias señales, utilizando el analizador lógico de este proyecto. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES		
Ref.	**Descripción**	**Cant.**
IC1 e IC2	Circuitos integrados 74LS241	2
IC3	Circuito integrado 555	1
IC4	Circuito integrado TL084	1
IC5	Circuito integrado TL071	1
REG1	Regulador fijo de +5V, 7805	1
TR1 a TR7	Transistores 2N3904	7
D1 a D4	Diodos rectificadores 1N4004	4
O0 a 04	Diodos LED de 3mm, rojos	5
I0 a I5 y LED3	Diodos LED de 3mm, verdes	6
LED1	Diodo LED de 5mm rojo	1
LED2	Diodo LED de 5mm verde	1
C1	Condensador electrolítico de 2200uF/25V	1
C2 a C9	Condensadores cerámicos de 0,1uF/50V	8
C11	Condensador electrolítico de 47uF/25V	1
C12 y C13	Condensadores electrolíticos de 10uF/25V	2
R1 a R6, R21 a R27	Resistencias de 330-1/4W	13
R7 a R20, R38 a R42,R44	Resistencias de 1k-1/4W	20
R28 a R32 y R43	Resistencias de 2k-1/4W	6
R33 a R37	Resistencias de 47k-1/4W	5
R45	Resistencia de 3.3k-1/4W	1
DIP	Interruptor tipo DIP de 4 posiciones	1
K-157	Circuito impreso CEKIT	1
K-157	Chasis metálico CEKIT	1
PROGRAMA K-157	Disquete grabado con programa K-157	1
	Conector DB25 hembra para impreso	1
	Conector CA/CC	1
	Conectores de tornillo de 3 pines	4
	Conector tipo cerca de 3 pines	1
	Bases para integrado de 20 pines	2
	Bases de 8 pines	2
	Base de 14 pines	1
	Cable computadora a *data switch*	1
	Adaptador de voltaje CA/CC	1
	Tornillos de 3 x15mm con sus tuercas	4
	Separadores plásticos	4
	Jumper	1

Paso 1

Instale y suelde los puentes de alambre y las resistencias R1 a R45.

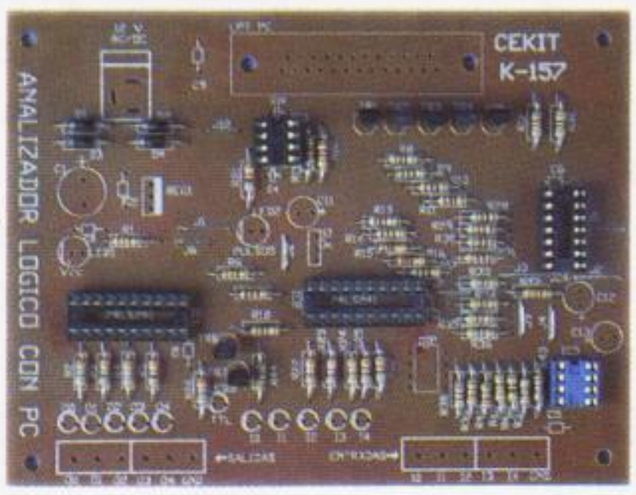
Figura 16.10



Paso 2

Luego instale y suelde los diodos rectificadores D1 a D4, las bases para los circuitos integrados, y los transistores NPN TR1 a TR7.

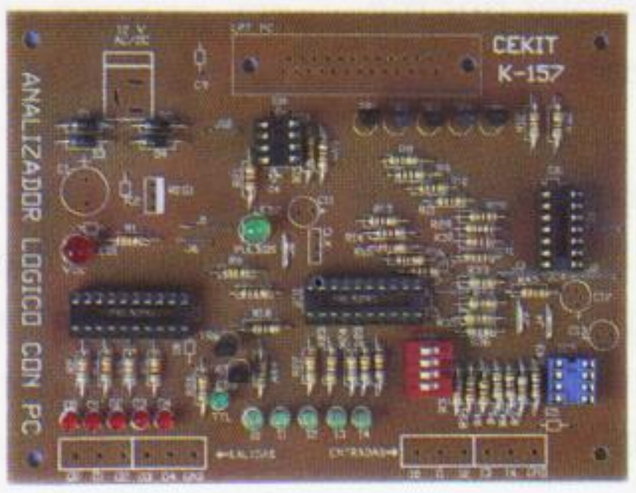
Figura 16.11



Paso 3

Instale y suelde los diodos LED de 3 y 5 mm y el interruptor de 4 posiciones tipo DIP.

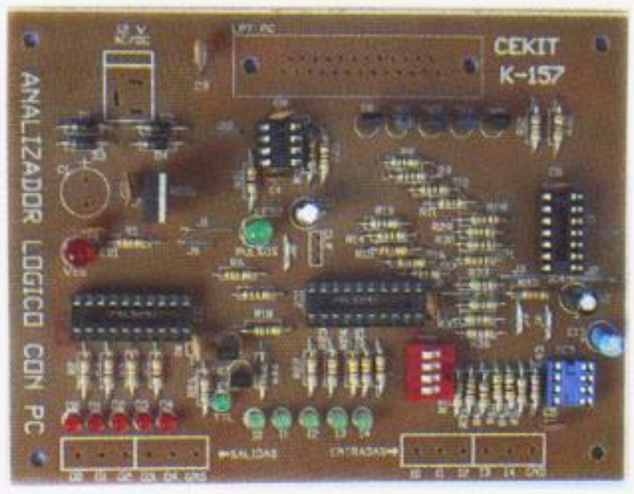
Figura 16.12



Paso 4

Instale y suelde los condensadores cerámicos C2 a C9, los condensadores electrolíticos C11 a C13 y el regulador de voltaje REG1.

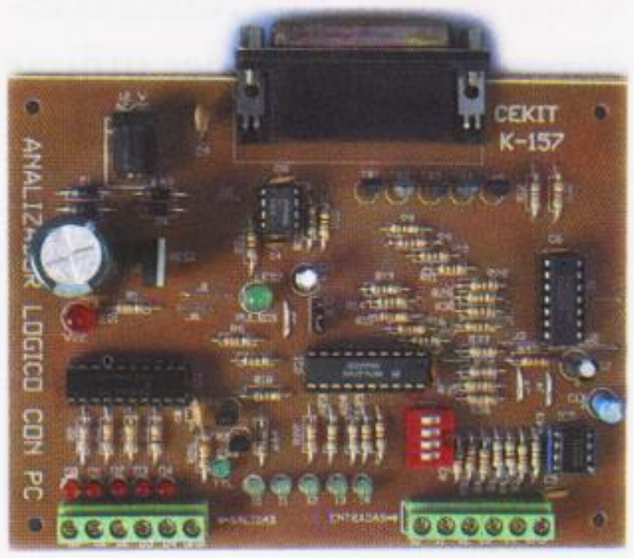
Figura 16.13



Paso 5

Aspecto final del circuito luego de instalar y soldar el condensador electrolítico C1, los conectores de entradas y salidas, el conector de alimentación CA/CC, el conector para el puerto de la computadora (DB-25), y el selector de frecuencia.

Figura 16.14



Paso 6. Pruebas finales

Para alimentar el circuito debe usar un adaptador que brinde un voltaje continuo entre 8 y 12 voltios. Cuando se alimenta la tarjeta por medio del conector CA/CC puede ejecutar el programa de la computadora y visualizar el comportamiento de las señales que se presentan por los pines de entrada. En este gráfico puede medir parámetros como el periodo y la frecuencia.

PROYECTO 17

CONTROL DIGITAL DE TEMPERATURA

En este proyecto usted va a construir un sistema que sirve como termómetro con indicación del valor medido en *displays* digitales. Al mismo tiempo, cumple una función de control, para la cual el usuario puede fácilmente graduar el punto de ajuste (*set point*) o valor de la temperatura en la cual se quiere hacer el control.

Características

- Visualización de la temperatura en dos *displays* de siete segmentos
- Graduación del punto de ajuste por medio de pulsadores para generar una señal de control
- Salida de control por medio de relé
- Rango de medidas desde 0º hasta 80º C

En general, un control de temperatura es un aparato que sirve para mantener estable, en un punto determinado, la temperatura de alguna sustancia o de algún sistema. Existen muchas aplicaciones para esta clase de aparatos; entre ellas se cuentan el control de temperatura de líquidos en laboratorios y procesos industriales; además del control de temperatura en cuartos o habitaciones especiales que deben mantener condiciones ambientales estables.

En la **figura 17.1** se muestra el diagrama en bloques del circuito, cada uno está formado por un grupo específico de componentes y cumple con una función determinada. A continuación, los describiremos y haremos una analogía con el correspondiente circuito utilizado, el cual se muestra en el diagrama esquemático que aparece en la **figura 17.2**.

Sensor de temperatura. Está conformado por un circuito integrado LM335, el cual entrega en su salida una señal que varía 10mV por cada grado Kelvin. Esto quiere decir que cuando la temperatura sea 0º C, entregará una señal de 2,73 voltios. Su apariencia es la de un transistor con encapsulado TO-92 y puede manejar perfectamente el rango de temperaturas requerido en este proyecto. Para utilizarlo, sólo se usan dos de sus pines, la salida que se conecta a la fuente de +5V con una resistencia (R1) y la tierra.

Convertidor A/D. En este proyecto, se utiliza un convertidor A/D de 8 *bits* con salida serial, referencia ADC0831. Como posee entrada diferencial, en la entrada positiva (Vin+) se conecta el sensor y en la entrada negativa (Vin-) se conecta un potenciómetro (R3) con el que se ajusta un voltaje de 2,73 voltios. Con esto se consigue que la salida de valor 00 del convertidor coincida con la temperatura de 0º C.

El voltaje de referencia se ha escogido como 1,27 voltios con el fin de detectar cambios de 5 mV en la señal de entrada, para ello se utiliza el potenciómetro R6. Así el termómetro puede tener una resolución de medio grado centígrado. Los pines del convertidor que se deben manejar desde la unidad de control son: CS (*chip select*), CLK (reloj) y DO (salida de datos). Éstos se han conectado a los pines RA0 a RA2 del microcontrolador PIC16F84.

Unidad de control. Está conformada por un microcontrolador PIC16F84 y su respectivo oscilador con cristal de 4 MHz. La razón para escoger este tipo de micro es que posee memoria de datos EEPROM interna lo que le permitirá almacenar el valor de la temperatura de control (*set point*), en una memoria no volátil. Además, se encarga de leer el dato del convertidor A/D, leer los parámetros de configuración, mostrar los datos correspondientes en el *display* digital y decidir si debe o no, accionar el relé de salida que actúa como elemento de control.

El programa que se graba en él le permite hacer todas las funciones. Como ejemplo, en la **figura 17.3** se muestra la parte del programa que lee el dato que entrega serialmente el convertidor A/D. En este caso, el micro genera las señales de CS (*chip select*) y de CLK (reloj) y lee el dato correspondiente a la conversión análogo a digital, que es entregado por el convertidor en su pin de salida DO.

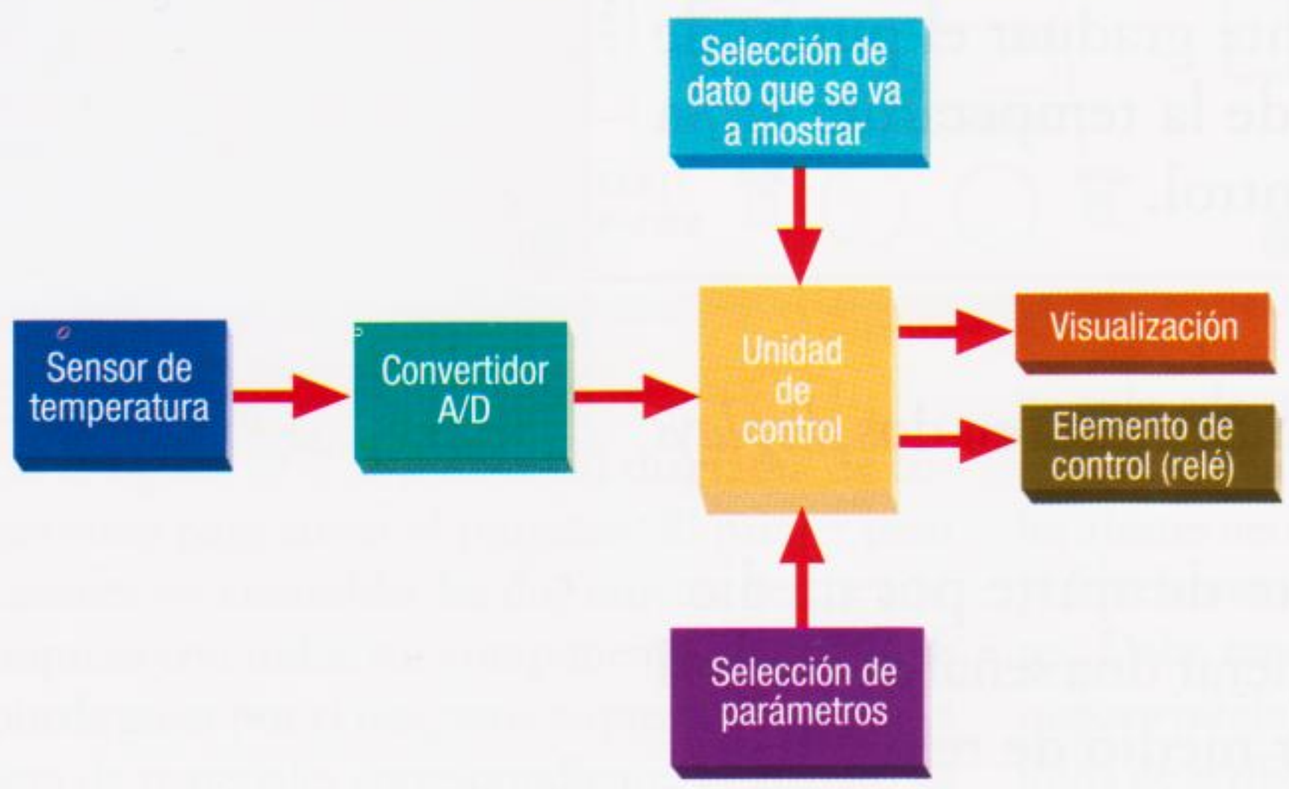


Figura 17.1 *Diagrama de bloques del sistema de control*

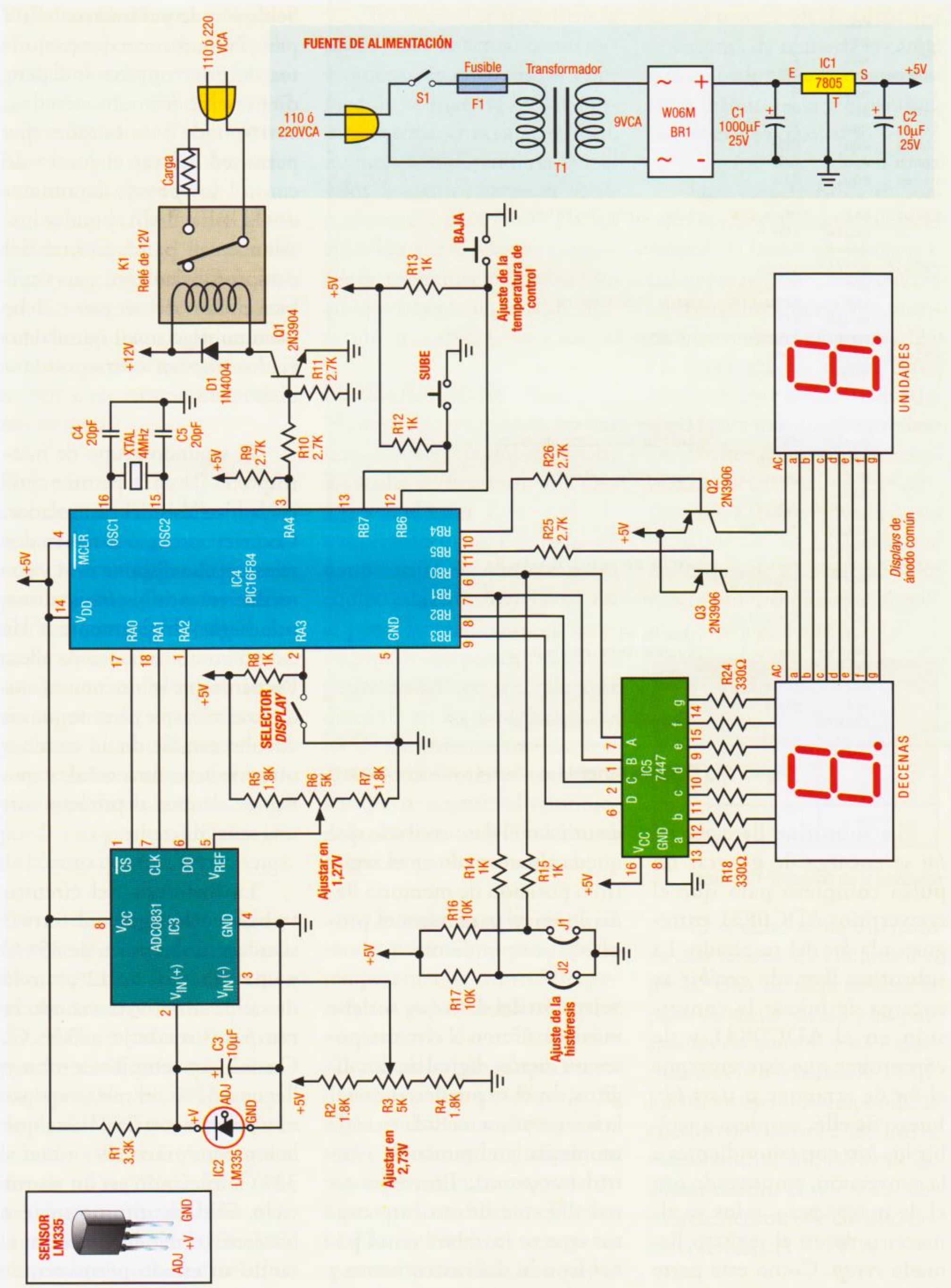


Figura 17.2 Diagrama esquemático del control digital de temperatura

```
CS      equ     0
CLK     equ     1
IN      equ     2
status  equ     03
c       equ     0
ptoa    equ     05
trisa   equ     85h
loops   equ     0ch
conta   equ     0eh
recep   equ     0fh

reloj   bsf     ptoa,CLK    ;sube línea de reloj
        movlw   1fh
        movwf   loops       ;retardo
reloj2  decfsz  loops
        goto    reloj2
        bcf     ptoa,CLK    ;baja línea de reloj
        movlw   1fh
        movwf   loops       ;retardo
reloj3  decfsz  loops
        goto    reloj3
        return

recibir clrf    recep       ;limpiar registro de Rx
        movlw   08h         ;inicia contador de bits
        movwf   conta
        bcf     ptoa,CS     ;baja chip select
        nop
        call    reloj
        btfsc   ptoa,IN     ;verifica dato en 0 lógico
        return
otrobit call    reloj       ;ciclo de reloj inicia Rx
        bsf     status,c
        btfss   ptoa,IN     ;lee dato del convertidor
        bcf     status,c
        rlf     recep,1     ;guarda el dato en recep
        decfsz  conta,1     ;cuando termine regresa
        goto    otrobit
        bsf     ptoa,CS
        bcf     ptoa,CLK
        return              ;retorna al programa principal
```

Figura 17.3 *Rutina del PIC16F84 para leer el convertidor ADC0831*

La subrutina llamada *reloj* se encarga de generar un pulso completo para que el convertidor ADC0831 entregue cada *bit* del resultado. La subrutina llamada ***recibir*** se encarga de iniciar la conversión en el ADC0831 y de comprobar que éste entregue el *bit* de arranque o *start bit*. Luego de ello, empieza a recibir los *bits* correspondientes a la conversión, empezando por el de mayor peso, y los va almacenando en el registro llamado ***recep***. Como esta parte del programa es una subrutina, el programa principal puede utilizar el dato recibido que queda almacenado en el registro o posición de memoria llamado ***recep*** para hacer el proceso correspondiente.

Selección del dato que se debe mostrar. Como el circuito posee un *display* digital de dos dígitos, en él se pueden mostrar la temperatura medida en cada momento y el punto de control (*set point*). Para ello usted dispone de un interruptor que se instalará en el panel frontal del instrumento y que le permitirá escoger el dato que desea ver.

Selección de parámetros. Esta parte la conforman dos conjuntos de interruptores independientes. El primero de ellos, corresponde a los botones que permiten ajustar el punto de control (*set point*), denominados SUBE y BAJA, y que se instalan en el panel frontal del equipo. Obviamente, para cambiar el dato del *set point*, debe primero seleccionar que el dato en los *displays* corresponda a dicho parámetro.

El segundo grupo de interruptores (J1 y J2), permite ajustar la histéresis del controlador. Como en este caso se tienen dos selectores, se dispone de 4 valores diferentes de histéresis correspondientes lógicamente a las cuatro combinaciones de ellos. Recuerde que la histéresis es una característica que permite que un circuito cambie de un estado a otro mediante una señal, y que regrese de éste al primero con una señal diferente.

La histéresis del circuito trabaja en la siguiente forma: si escoge un *set point* de 35° C y una histéresis de ±2°, el relé de salida se activará cuando la temperatura baje a 33° C. Cuando ésta empiece a subir y llegue a 37°C, el relé se apagará y permanecerá así hasta que la temperatura vuelva a caer a 33° C, iniciando así un nuevo ciclo. Si el circuito no tuviera histéresis, el relé estaría conmutando su estado permanentemente, disminuyendo considerablemente su vida útil.

Visualización. Como ya se dijo, el sistema posee un *display* digital de dos dígitos, conformado por un decodificador 7447 y dos *displays* de 7 segmentos. Esto implica que los datos deben ser mostrados en forma multiplexada, es decir, se envía al decodificador el dato de un *display* y se habilita el transistor que le corresponde. Todo este proceso se hace rápidamente para que parezca que ambos elementos están encendidos al mismo tiempo.

Elemento de control. En este caso, se utiliza un relé como elemento de salida ya que éste permite conectar cargas de gran capacidad como calentadores, ventiladores y en general, algún elemento que permita cambiar las condiciones ambientales del sitio al que se le controla la temperatura. Como dispone de los contactos normalmente abiertos y normalmente cerrados, usted puede escoger la configuración de la carga que más le convenga.

Fuente de alimentación. El sistema se alimenta de la red pública de 110 ó 220 VCA, posee un transformador que reduce dicha tensión a 9VCA, un puente rectificador que permite obtener un voltaje aproximado de 11VCC para alimentar la bobina del relé, y un regulador de +5V (7805). Este último, le permite alimentar toda la parte digital del sistema, incluyendo el microcontrolador, el convertidor A/D, los *displays* y demás elementos.

También el aparato tiene la posibilidad de actuar como termómetro, en cuyo caso el valor medido se muestra en los mismos *displays*. Otra característica, no menos importante, es que el valor de control (*set point*) se almacena en una memoria no volátil lo que le permite conservar la programación del equipo aunque haya retirado la alimentación del sistema.

Funcionamiento

El control de temperatura posee una pantalla con dos dígitos en los cuales se muestran, de a uno a la vez, el valor de control y la temperatura medida. El rango de temperatura en que trabaja el equipo está entre 0º y 80º C. En el panel frontal existe un interruptor de dos posiciones que le permite establecer si el valor que desea ver en los *displays* es el valor de temperatura medido o el de control. En caso de seleccionar que se muestre el valor medido, los botones pulsadores no hacen ninguna operación.

Cuando se selecciona el modo para ver el valor de la temperatura de control (*set point*), los pulsadores sirven para aumentar o disminuir dicho valor, cambio que se nota inmediatamente en los *displays*. Si algún pulsador se deja presionado, el número del *display* se incrementará automáticamente hasta que usted decida liberarlo. Durante todo el tiempo en que está encendido el equipo, se están efectuando constantes medidas y verificando si se debe activar o no el relé de salida que se encarga de manejar la carga, o el actuador que hace aumentar o disminuir la temperatura del objeto bajo control.

Una característica importante es que puede seleccionar el valor de la histéresis del controlador; para ello se han dispuesto en el circuito impreso dos puentes (*jumpers*), denominados J2 y J1, mediante los cuales usted puede escoger el valor más adecuado para su situación particular. Dichos valores están comprendidos entre ±1º y ±4º centígrados. El valor específico está dado por la **tabla 17.1** en la cual un 0 equivale a un puente en su sitio, y un 1 indica que el mismo no está presente:

J2	J1	Histéresis
0	0	±1°
0	1	±2°
1	0	±3°
1	1	±4°

Tabla 17.1 *Selección de la histéresis*

El circuito se ha diseñado en dos tarjetas de circuito impreso. Una de ellas incluye la parte de visualización y los pulsadores de control y está marcada con la referencia K-191B. La otra tarjeta corresponde al circuito de control e incluye el microcontrolador, la fuente de alimentación, el relé y los demás elementos; este circuito tiene la referencia K-191A.

Circuito de control. En la **figura 17.4** se muestra el circuito im-

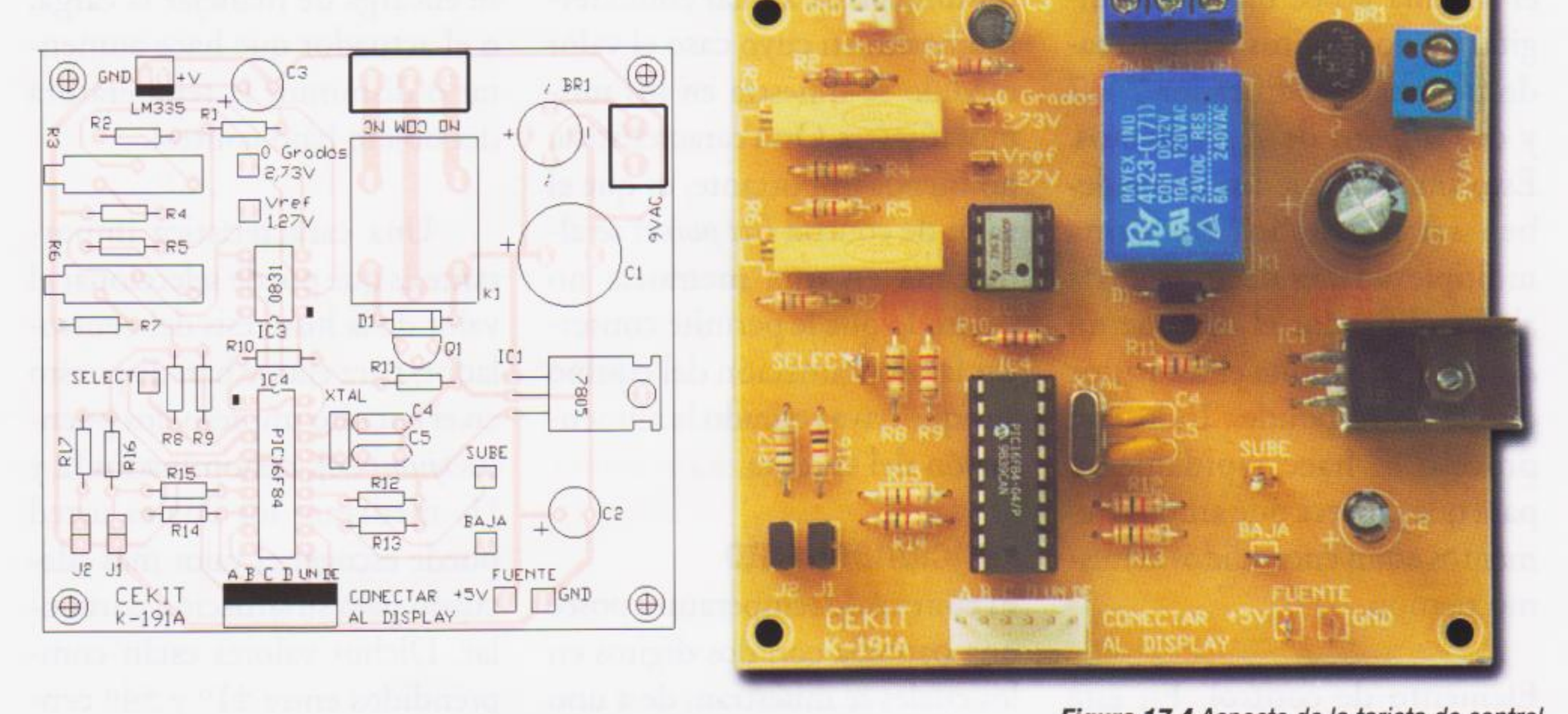


Figura 17.4 *Aspecto de la tarjeta de control*

preso de la tarjeta de control del equipo; en ella se han montado el microcontrolador, la parte de acondicionamiento y tratamiento de la señal análoga del sensor de temperatura, el relé que maneja la carga de salida y la fuente de alimentación del sistema.

El microcontrolador por su parte, trabaja con un cristal de 4 MHz que le permite una buena estabilidad para efectos del cálculo de los tiempos. Además, como se dijo anteriormente, este sistema puede almacenar en memoria el valor de la temperatura de control; para ello se utiliza la memoria EEPROM de datos que posee el PIC16F84, la cual permite almacenar hasta 64 bytes si se requiere. Esta característica hace que se pueda retirar la fuente de alimentación del sistema sin que se pierda dicha información. El programa con que se graba el microcontrolador permite además cumplir con todas las demás funciones que se describen.

También es importante el tipo de conexión del sensor de temperatura ya que en el circuito impreso se ha colocado un conector en línea de dos pines, el cual sirve para llevar las señales correspondientes hasta el conector (*plug*) monofónico que se ubica en el *chasis* (parte posterior), para que el sensor de temperatura se pueda conectar y desconectar fácilmente desde el exterior del equipo con un conector (*jack*) monofónico.

Debe tener en cuenta que el sensor se conecta en forma de sonda, utilizando cable blindado para sus dos señales y un poco de espagueti termoencogible para aislar sus pines y proteger su encapsulado. Por su parte, el relé que maneja la carga tiene un conector de tornillo en el circuito impreso, en el cual se conectan los cables de salida. En el *chasis* del equipo, en la parte posterior, se coloca una regleta de tres terminales a través de la cual están disponibles los contactos del relé para que usted efectúe fácilmente la conexión que más le convenga.

Las señales de control que van hacia la tarjeta de visualización se llevan a través de un cable plano (*ribbon*), el cual utiliza un conector en línea de 6 pines para que el sistema pueda ser fácilmente armado y desarmado.

La conexión de los botones pulsadores, del interruptor de selección del valor mostrado y de la fuente de alimentación del *display*, se hacen mediante cables que van conectados entre los dos circuitos impresos, en los puntos marcados con el nombre de la señal correspondiente (SUBE, BAJA, SELECT, +5V y GND).

El convertidor análogo a digital ADC0831 tiene dos voltajes

asociados que deben ser calibrados por usted para obtener un correcto funcionamiento del equipo. El primero corresponde al voltaje negativo de entrada del convertidor (Vin-), mediante el cual se hace la compensación para ajuste de 0º C del sensor LM335. Este paso es necesario ya que, como es sabido, el sensor LM335 entrega un valor de 2,73 voltios cuando la temperatura es de 0º centígrados. Para ajustar este valor utilice el potenciómetro (*trimmer*) marcado R3 y la medida correspondiente se puede hacer en el punto de prueba marcado: *0 Grados / 2,73V*.

El otro voltaje que debe ser ajustado corresponde al voltaje de referencia del convertidor, el cual se ha establecido en 1,27 voltios, lo que permite detectar cambios de 5mV en la señal de entrada, ya que el ADC0831 tiene una resolución de 8 *bits*. El ajuste se hace mediante el potenciómetro (*trimmer*) marcado R6; la medida correspondiente se puede hacer en el punto de prueba marcado: *Vref / 1,27V*.

La fuente de alimentación del sistema es muy simple, con ella se obtiene un voltaje cercano a los 11VCC con el cual se alimenta la bobina del relé y a su vez genera los +5V para alimentar toda la parte digital del sistema. En la entrada del circuito se debe conectar un transformador que permita obtener 9VCA a partir de la línea de 110 ó 220 VCA.

Circuito de visualización. El circuito impreso que contiene los *displays* y los interruptores se debe ubicar en el panel frontal del equipo, de tal forma que usted pueda tener fácil acceso a los controles y pueda observar los valores medidos o de control, de forma inmediata. Como se dijo en la primera parte de este proyecto, se utilizó un sistema de *display* multiplexado para el despliegue de la información. Este procedimiento permite ahorrar en materiales y espacio ya que sólo se utiliza un decodificador 7447, además que se disminuye el consumo de energía debido a que en realidad sólo se activa un *display* en cada momento. En la práctica se ha encontrado que activar cada *display* durante tres milisegundos produce un buen resultado.

Los pulsadores que permiten ajustar el tiempo en los *displays* son del tipo normalmente abierto; éstos se conectan hasta el circuito de control mediante cables individuales como se dijo anteriormente. Igual sucede con el interruptor de selección de dato que se va a mostrar. El cable plano que tiene las señales o datos que se muestran en los *displays* se conecta en su respectivo sitio con puntos de soldadura, ya que en el otro extremo del cable se instaló un conector en línea lo que permite separar las dos tarjetas fácilmente. En la **figura 17.5** se muestra el circuito impreso de la tarjeta de visualización y en la **figura 17.6** el aspecto final del circuito ensamblado.

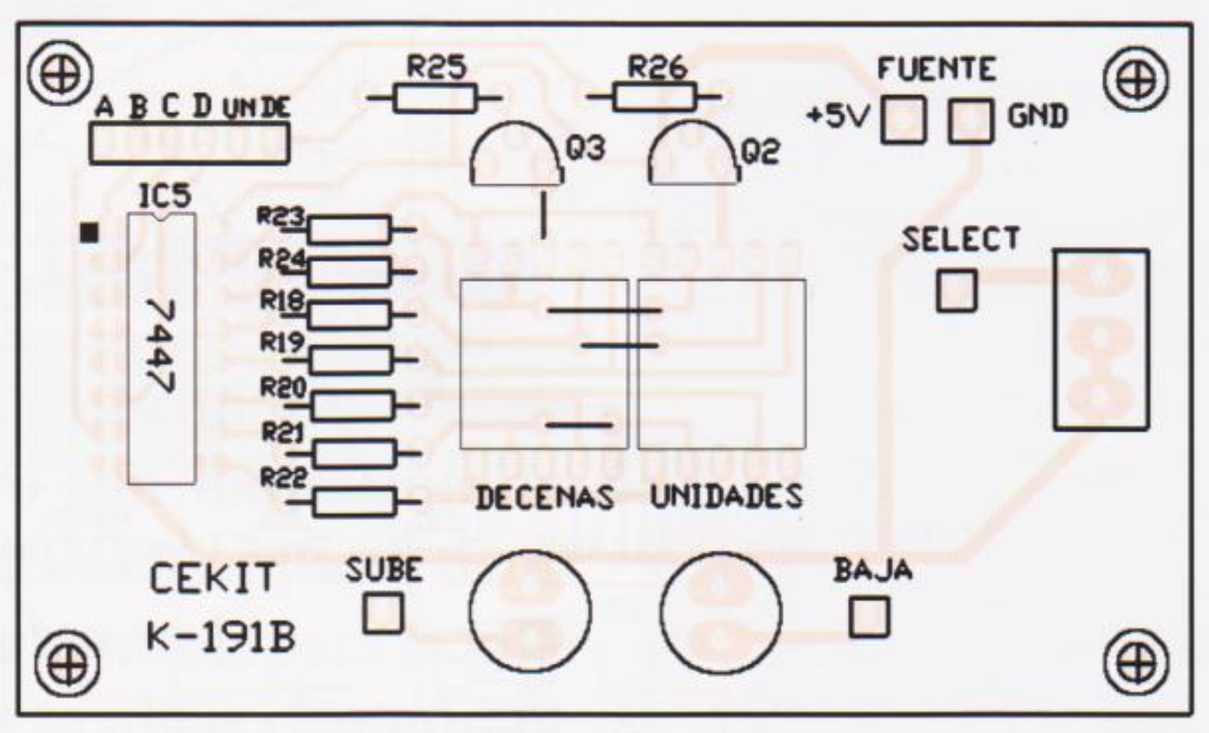


Figura 17.5 *Guía de montaje del circuito de visualización*

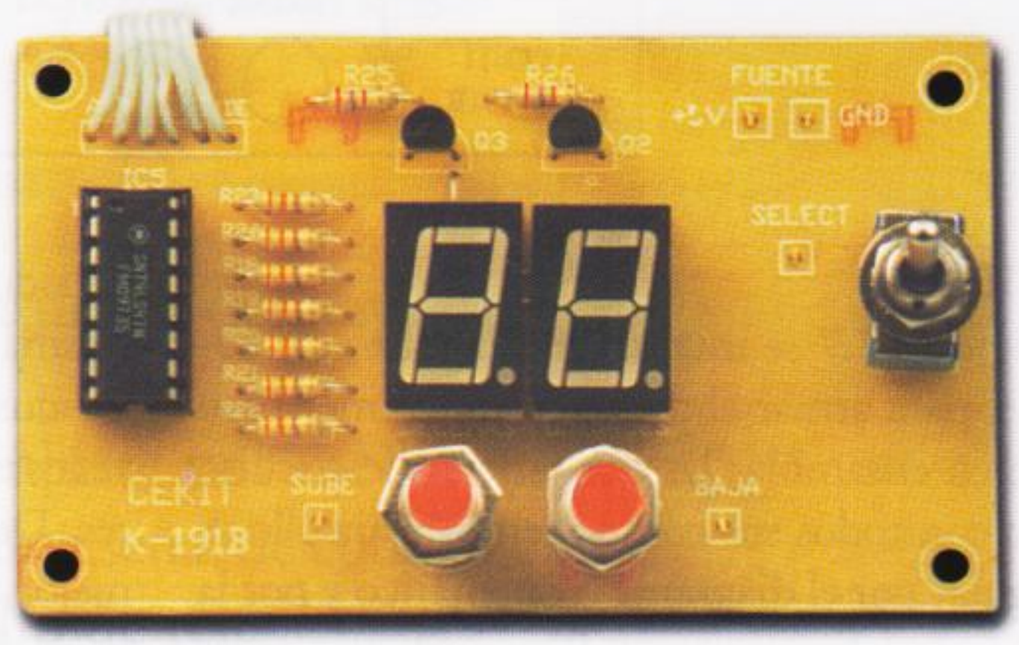


Figura 17.6 *Tarjeta de visualización totalmente ensamblada*

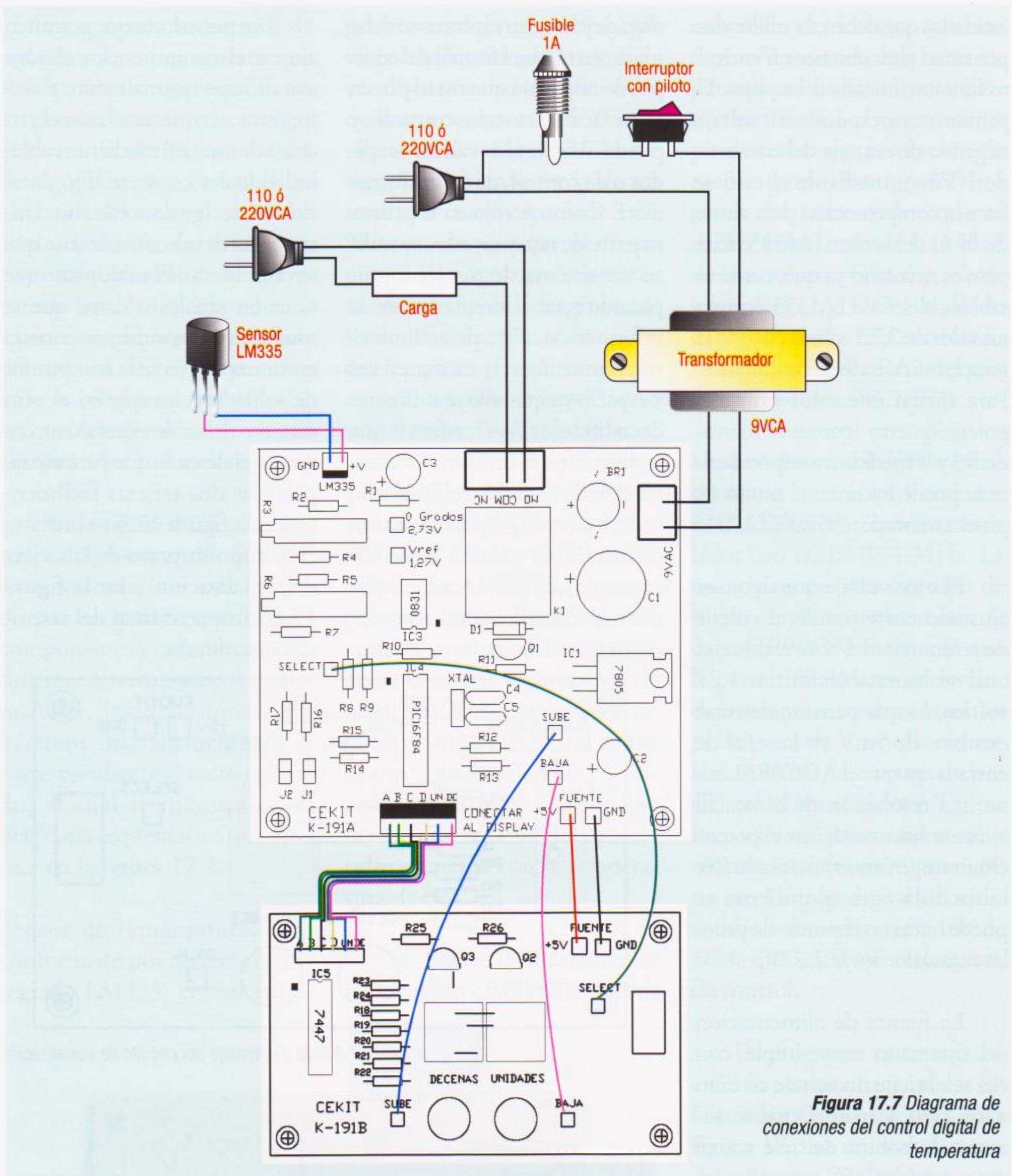


Figura 17.7 *Diagrama de conexiones del control digital de temperatura*

Ensamblaje

En la **figura** 17.7 se muestra el diagrama de conexiones para armar el proyecto. El primer paso consiste en ensamblar las dos tarjetas de circuito impreso con todos sus componentes, para ello se puede guiar por el diagrama esquemático y por la lista de materiales correspondiente. El aparato va montado dentro de una caja metálica (*chasis*) que ha sido adecuada para tal fin. En el frente de la misma se han dispuesto todas las perforaciones y los ajustes necesarios para fijar la tarjeta que incluye los elementos de visualización y los interruptores. Debe tener en cuenta que para obtener una mejor visibilidad es conveniente emplear una lámina de acrílico ahumado enfrente de los *displays*; para fijarla se puede utilizar pegante instantáneo.

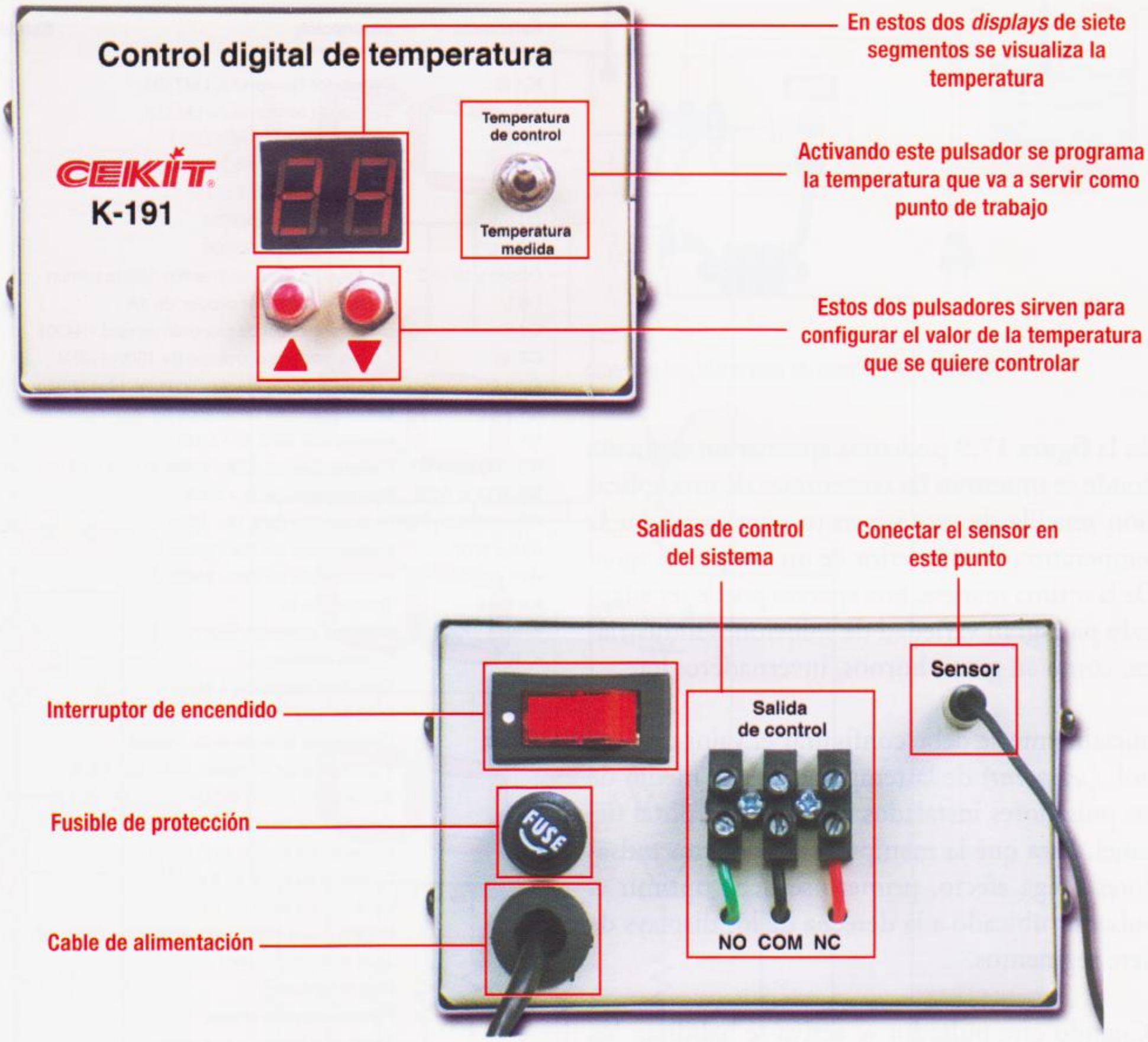


Figura 17.8 *Paneles frontal y trasero del control de temperatura*

En la parte trasera del *chasis* se fija un conector o regleta con bornes para conectar la carga que va a ser manejada por el relé. En la **figura 17.8** se muestra el diseño de los paneles frontal y trasero del control de temperatura.

Durante el ensamblaje, debe tener cuidado en soldar muy bien y evitar cortocircuitos entre los puntos que se encuentren muy cercanos. Para fijar la tarjeta de control dentro del *chasis*, que va sobre el piso del mismo, debe utilizar tornillos con tuerca y separadores plásticos de al menos 8 mm, esto con el fin de evitar que la parte inferior de la tarjeta toque la lámina metálica, lo que podría causar cortocircuitos.

La tarjeta de visualización se debe fijar al panel frontal mediante las tuercas que poseen los interruptores del sistema.

La apariencia física de este proyecto (forma de panel) se ha sometido a un proceso de diseño cuyo objetivo es presentar un producto de características comerciales que pueda competir en cuanto a calidad y precio con modelos similares disponibles en el mercado. En las figuras puede observar la facilidad de acceso y la disposición estratégica dentro del *chasis* que poseen los distintos elementos de interacción con el usuario.

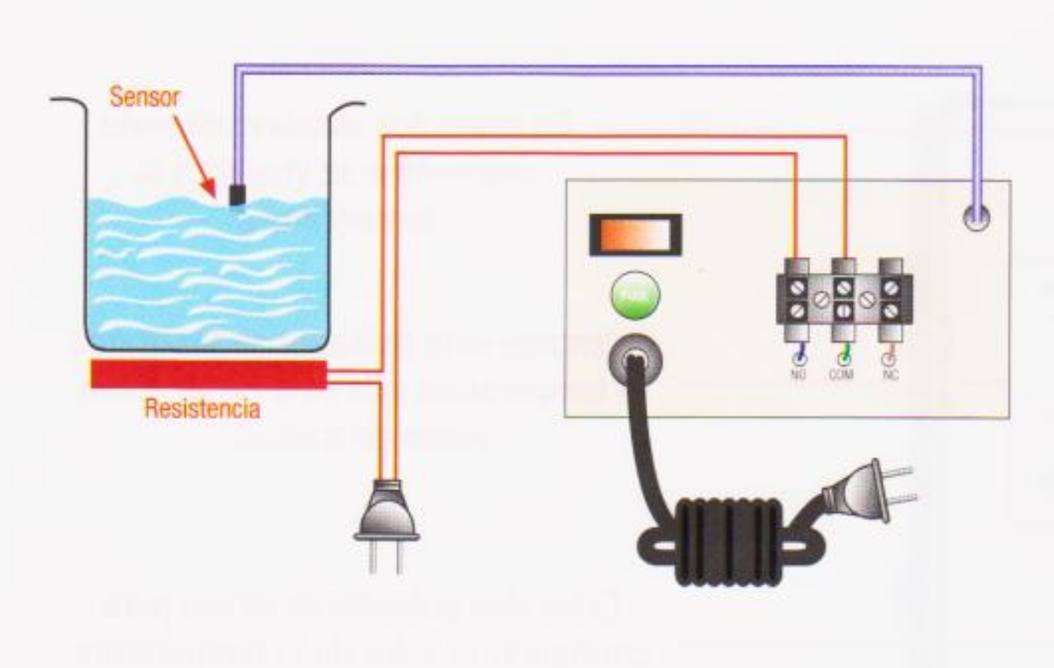


Figura 17.9 *Ejemplo de aplicación del proyecto*

En la **figura 17.9** podemos apreciar un esquema donde se muestran las conexiones de una aplicación sencilla de este proyecto para controlar la temperatura en el interior de un tanque de agua. De la misma manera, este aparato puede ser adaptado para gran variedad de soluciones industriales, como en gases, hornos, invernaderos, etc.

Inicialmente se debe configurar el valor de control (*set-point*) de la temperatura por medio de los pulsadores instalados en la parte frontal del panel. Para que la manipulación de estos pulsadores tenga efecto, primero se debe oprimir el pulsador ubicado a la derecha de los displays de siete segmentos.

Cuando este pulsador se activa se habilitan los pulsadores de calibración y ese valor ajustado queda almacenado en la memoria. El sensor de temperatura debidamente protegido con la cinta termoencogible se instala en el interior de tanque para tomar la lectura.

Cuando la temperatura leída alcanza el valor programado, origina una salida hacia un circuito de potencia que se encarga de llevar a cabo la acción de control, que en este caso se trata de un calentador ubicado en la parte inferior del tanque. Este tipo de control se suele denominar ON-OFF, ya que se limita a activar o desactivar el elemento final de control. Cuando el dispositivo cierra los contactos del relé, se cierra un circuito de 120 o 220 voltios CA, en donde se encuentra el calentador conectado en serie.

LISTA DE MATERIALES

Referencia	Descripción	Cantidad
IC1	Regulador fijo de +5V, LM7805	1
IC2	Sensor de temperatura LM335	1
IC3	Circuito integrado ADC0831	1
IC4	Microcontrolador PIC16F84 grabado	1
IC5	Circuito integrado 7447	1
Q1	Transistor NPN 2N3904	1
Q2 y Q3	Transistor PNP 2N3906	2
DISP1 y DISP2	Displays de siete segmentos ánodo común	2
BR1	Puente rectificador circular de 1A	1
D1	Diodo rectificador de propósito general 1N4004	1
C1	Condensador electrolítico de 1000µF/25V	1
C2 y C3	Condensadores electrolíticos de 10µF/25V	2
C4 y C5	Condensadores cerámicos de 20pF/25V	2
R1	Resistencia de 3,3k - 1/4W	1
R2, R4, R5 y R7	Resistencias de 1,8k - 1/4W	4
R8, R12 a R15	Resistencias de 1k - 1/4W	5
R9 a R11, R25 y R26	Resistencias de 2,7k - 1/4W	5
R16 y R17	Resistencias de 10k - 1/4W	2
R18 a R24	Resistencias de 330 - 1/4W	7
R3 y R6	Trimmer de 5k	2
K-191A y K-191B	Circuitos impresos CEKIT	2
K-191	Chasis metálico	1
XTAL	Cristal de cuarzo de 4 MHz	1
K1	Relé de 12V	1
J1 y J2	Conectores tipo cerca de 2 pines	2
	Transformador Primario: 110 ó 220 VCA	
	Secundario: 9VCA M-504	1
	Conector de tornillo de 2 pines	1
	Conector de tornillo de 3 pines	1
	Conector en línea de 2 pines	1
	Conector en línea de 6 pines	1
	Regleta de conexión negra de 3 pines, pequeña	1
	Jack monofónico para chasis	1
	Plug monofónico	1
	Pulsadores para chasis	2
	Interruptor balancín con piloto	1
	Interruptor de codillo de 2 posiciones	1
	Base para integrado de 8 pines	1
	Base para integrado de 16 pines	1
	Base para integrado de 18 pines	1
	Jumpers	2
	Terminales para circuito impreso (espadines)	12
	Disipador TO-220	1
	Cable blindado monofónico (50 cm)	1
	Cable ribbon de 6 hilos (40 cm)	1
	Cable vehículo calibre 20 AWG rojo (1 m)	1
	Cable vehículo calibre 20 AWG negro (1 m)	1
	Separadores plásticos de 8 mm	4
	Portafusible corto para chasis	1
	Fusible corto de 1A	1
	Pasacable de caucho	1
	Lámina de acrílico ahumado de 3 x 3,5 cm	1
	Tornillo milimétrico 3x15 con tuerca	8
	Tornillo milimétrico 3x7 con tuerca	1
	Tornillo para lámina 3x7	8
	Termoencogible (2 cm)	1
	Cable de potencia con enchufe	1
	Soldadura (1 m)	1

Ensamblaje del proyecto

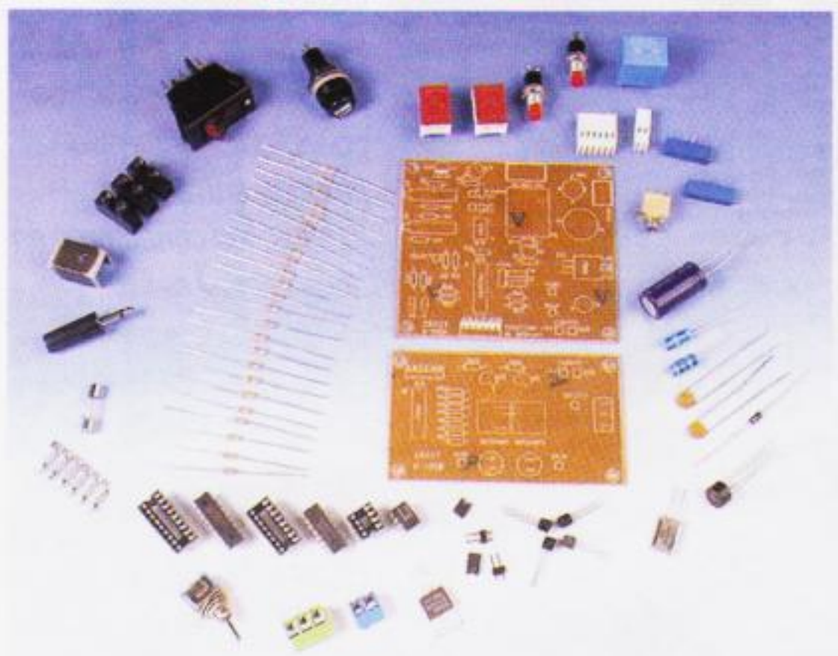
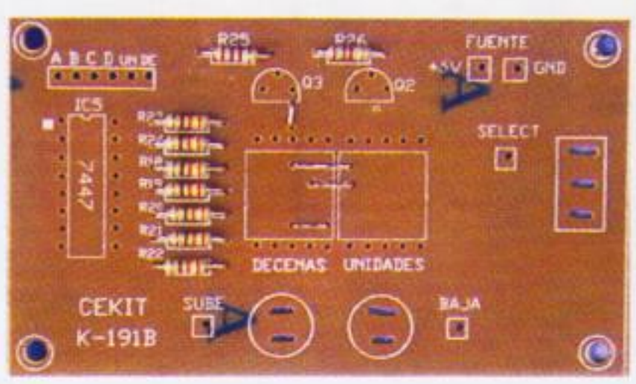
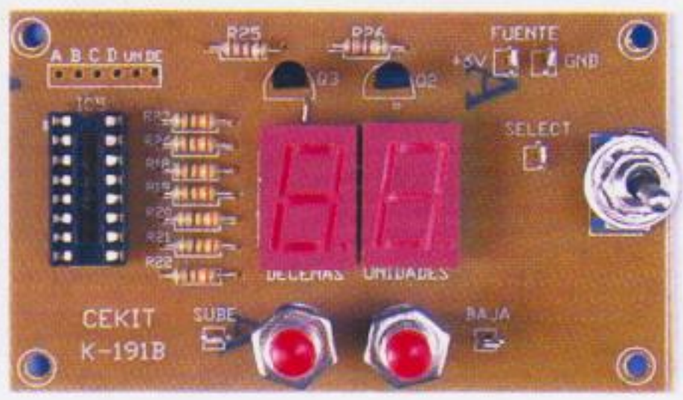


Figura 17.10 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el Control digital de temperatura CEKIT K-191.*

La medición y el control de la temperatura son de gran importancia debido a sus frecuentes aplicaciones en procesos industriales y residenciales. El control digital de temperatura de CEKIT K-191 puede usarse en el control de temperaturas entre 0 y 180 grados centígrados. Para el ensamblaje de las tarjetas tenga en cuenta los siguientes pasos:

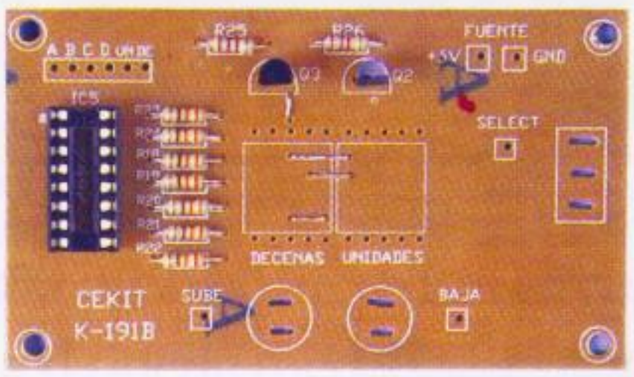
Paso 1
Instale y suelde en la tarjeta de visualización K-191B los puentes de alambre y las resistencias R18 a R26.
Figura 17.11

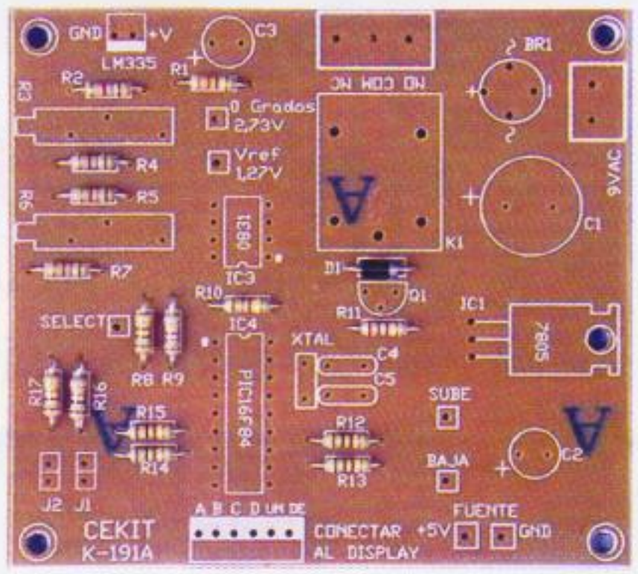




Paso 4
Instale y suelde por último, el interruptor selector y los dos pulsadores para subir y bajar el valor de la temperatura.
Figura 17.14

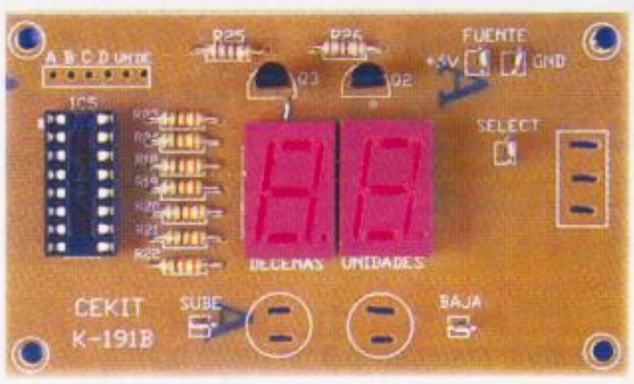
Paso 2
Luego instale y suelde la base para el circuito integrado IC5 y los transistores PNP Q2 y Q3.
Figura 17.12

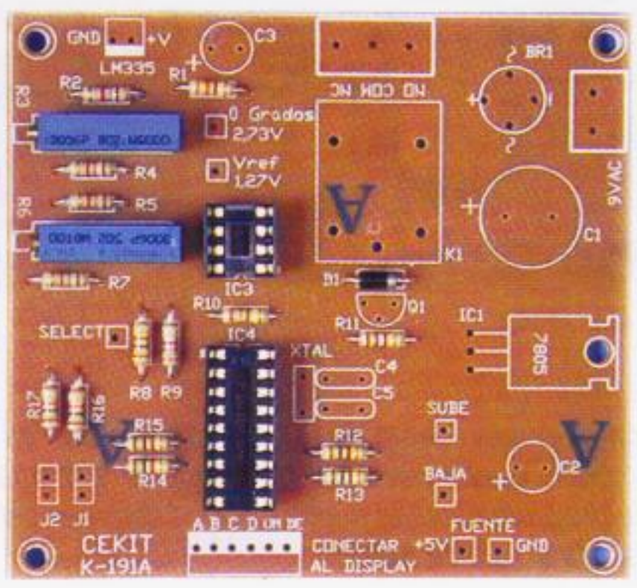




Paso 5
Instale y suelde en la tarjeta de control K-191A los puentes de alambre, las resistencias R1,R2, R4, R5 y R7 a R17, y el diodo rectificador D1.
Figura 17.15

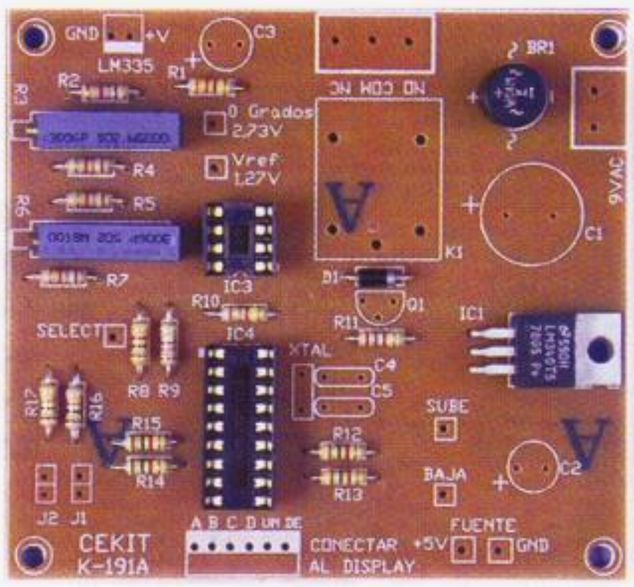
Paso 3
Instale y suelde los displays de ánodo común para las unidades y las decenas y los terminales para circuito impreso (espadines)
Figura 17.13





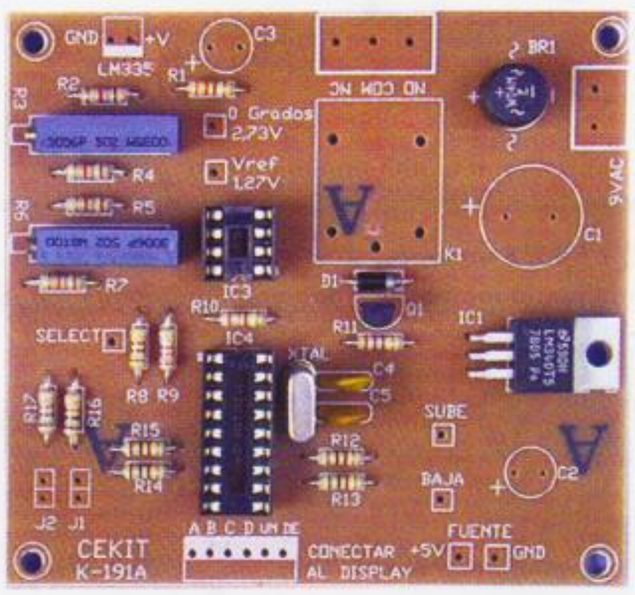
Paso 6
Luego instale y suelde las bases para los circuitos integrados IC3 e IC4 y las resistencias variables R3 y R6.
Figura 17.16

Paso 7
Instale y suelde el puente rectificador BR1, doble los terminales del regulador de voltaje IC1 y asegúrelo en el lugar indicado por la guía de ensamble.
Figura 17.17



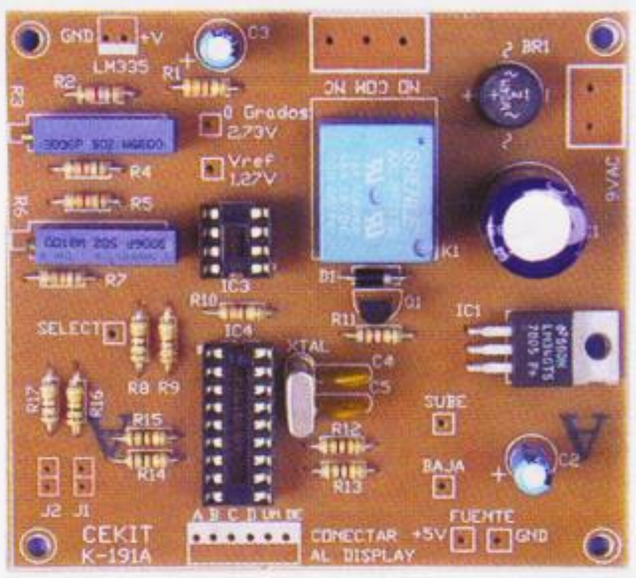
Paso 10
Instale y suelde los terminales para el circuito impreso (espadines), los conectores de tornillo para las entradas y salidas, los conectores en línea para el sensor y la tarjeta de visualización; asegure el disipador de IC1 con un tornillo milimétrico de 3X7 con tuerca.
Figura 17.20

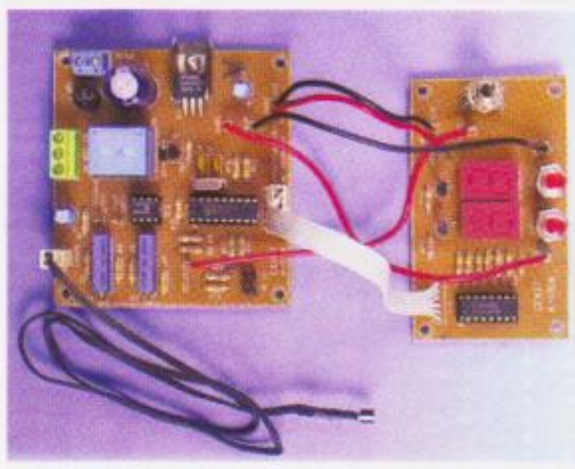
Paso 8
Instale y suelde los condensadores cerámicos C4 y C5, el transistor Q1 y el cristal de cuarzo de 4 MHz.
Figura 17.18



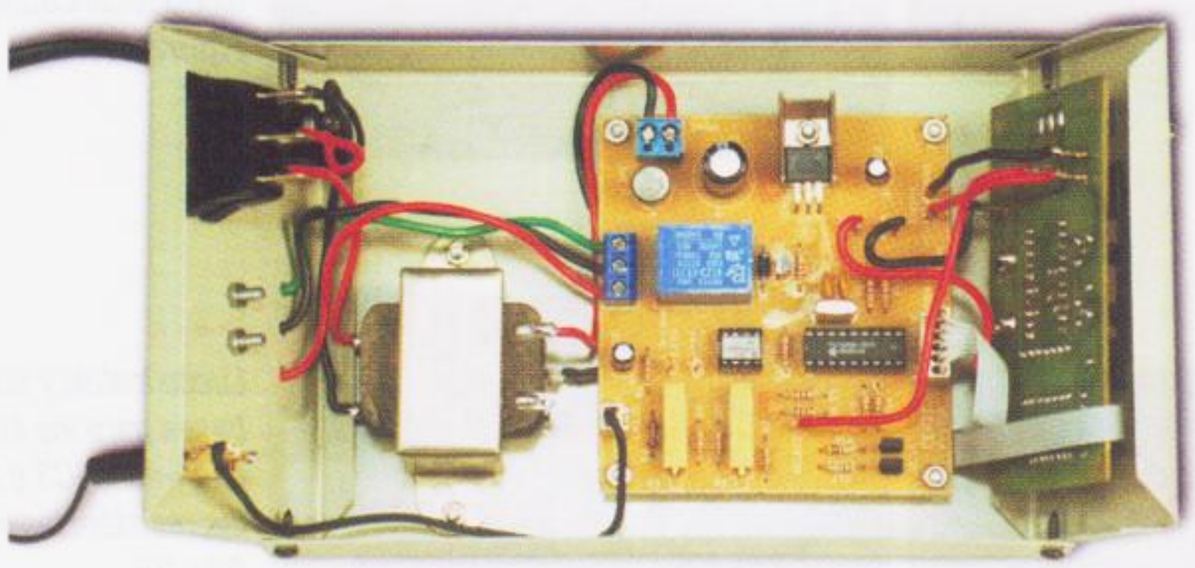
Paso 11
Para construir la sonda de medida de temperatura, se usa el sensor de temperatura LM335 y se conectan solamente dos de sus terminales +V y GND. Suelde los cables a los terminales del sensor y cúbralos con espagueti termoencogible, tal como se aprecia en la figura.
Figura 17.21

Paso 9
Instale y suelde el relé de 12V K1 y los condensadores electrolíticos C1, C2 y C3.
Figura 17.19





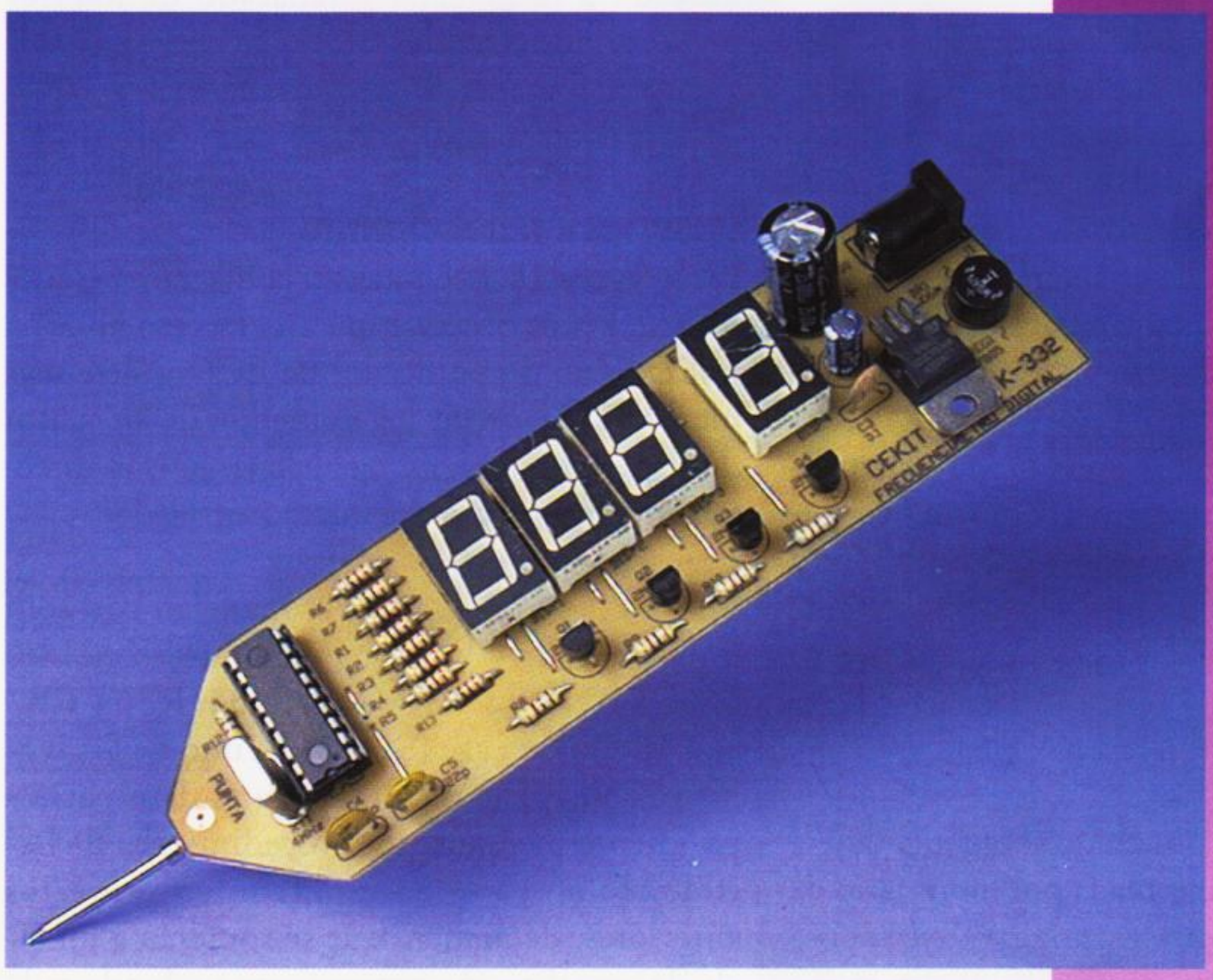
Paso 12
Aspecto final de las tarjetas electrónicas de los circuitos de visualización y control totalmente ensambladas, con la sonda de temperatura y con las conexiones respectivas entre ellas, tal como se ilustra en la figura 17.7 (diagrama de conexiones).
Figura 17.22



Paso 13
Ubicación y conexión de las tarjetas electrónicas y los otros elementos (transformador, interruptor, portafusible, regleta de conexión y entradas de potencia y el sensor) dentro del chasis .
Figura 17.23

Paso 14. Pruebas finales
Este proyecto cuenta con su propia fuente de CC, por lo tanto solo es necesario conectarlo a un toma normal de 110 o 220 VCA. Cuando se alimenta el módulo puede visualizar la lectura de la temperatura leída con el sensor, si éste se ha instalado correctamente. Al oprimir el pulsador de programación, el número en los displays debe modificarse. Inicialmente programe la temperatura embiente para verificar que se dispara el relé de salida.

PROYECTO 18



FRECUENCÍMETRO DIGITAL

Este proyecto es una herramienta de laboratorio muy importante para probar diferentes circuitos. Con este dispositivo usted puede medir la frecuencia de señales digitales y visualizarla en *displays* de siete segmentos.

Características

- Frecuencímetro auto-rango
- Visualización en cuatro *displays* de siete segmentos
- Rango de frecuencias de 0 Hz a 50 MHz
- Instrumento portátil
- Tamaño reducido
- Bajo costo

Una de las características más importantes de las señales digitales es su frecuencia. Dentro de un sistema digital se generan muchas señales en donde el manejo de este parámetro requiere de gran precisión y exactitud.

De no ser así se puede caer en fallas de sincronización, errores en la generación de tiempos, etc., que pueden afectar el buen funcionamiento de una tarjeta o equipo. Por estas razones, en el campo del desarrollo electrónico es muy importante el seguimiento de esta variable en las distintas etapas de los circuitos para detectar posibles fallas o imprecisiones.

Teniendo en cuenta lo anterior, es fundamental contar con una herramienta como un frecuencímetro dentro de su laboratorio de electrónica digital y microcontroladores; ese es el objeto de incorporar la construcción del presente proyecto dentro de este curso.

Funcionamiento

Este proyecto se ha construido sobre una tarjeta diseñada para ser manipulada por un usuario de manera similar a como lo haría con una punta lógica. Esta característica, sumada al alto rango de lecturas que ofrece, hacen de este instrumento un producto que supera a muchos sistemas similares que se encuentran en el mercado a un costo más alto.

Este equipo permite leer frecuencias comprendidas en el rango de 0Hz hasta 50MHz, y simplemente utiliza un microcontrolador PIC trabajando con un cristal de 4 MHz. El circuito impreso se ha diseñado en forma de "lapicero" para facilitar su manipulación por parte del usuario y en él se destacan los cuatro *displays* de siete segmentos que utilizará para visualizar el resultado de sus mediciones.

Lectura de los *displays*

Como sólo se cuenta con cuatro *displays* para visualizar la lectura, se ha adoptado una notación de ingeniería para mostrar el resultado final. De esta manera los tres primeros dígitos mostrarán un dato que debe ser elevado a la potencia de 10 indicada en el cuarto *display*, es decir: A.BC * 10 ^D. Debido a esta distribución, el *display* del exponente se ha ubicado a la derecha y separado de los tres primeros dentro del circuito impreso. Por ejemplo, una lectura típica sería $1{,}20 \times 10^3$ lo que equivale a 1.200 Hz.

Descripción del circuito

En la **figura 18.1** se muestra el diagrama esquemático. Puede observar que en este caso no se ha utilizado un decodificador de BCD a siete segmentos, cada uno de los segmentos de los cuatro *displays* conectados en paralelo se activan directamente por un pin del microcontrolador.

Los cuatros *displays* se manejan con solo once pines de salida y cuatro transistores; los pines RB1, RB2, RB3, RB4, RB5, RB6 y RB7 se utilizan para alimentar cada uno de los segmentos. Cada *display* muestra el dato correspondiente mediante un procedimiento de barrido, como ya se ha explicado en proyectos anteriores, de manera que se someten a un ciclo de actualización permanente en donde cada uno permanece activado durante 62 us.

Cada *display* posee su propio transistor que permite su activación, siendo estos manejados por los pines RA0, RA1, RA2 y RA3.

El microcontrolador funciona con un cristal de 4 MHz y la entrada de frecuencia que se va a medir corresponde al pin RA4, que como hemos estudiado en distintas lecciones, corresponde a la entrada para el TMR0.

En la **figura 18.2** puede apreciar la guía de montaje donde se señala la ubicación de todos los componentes y que puede utilizar para ensamblar la tarjeta.

Como puede apreciar la entrada de frecuencia tiene también conexión con el pin RB0, el

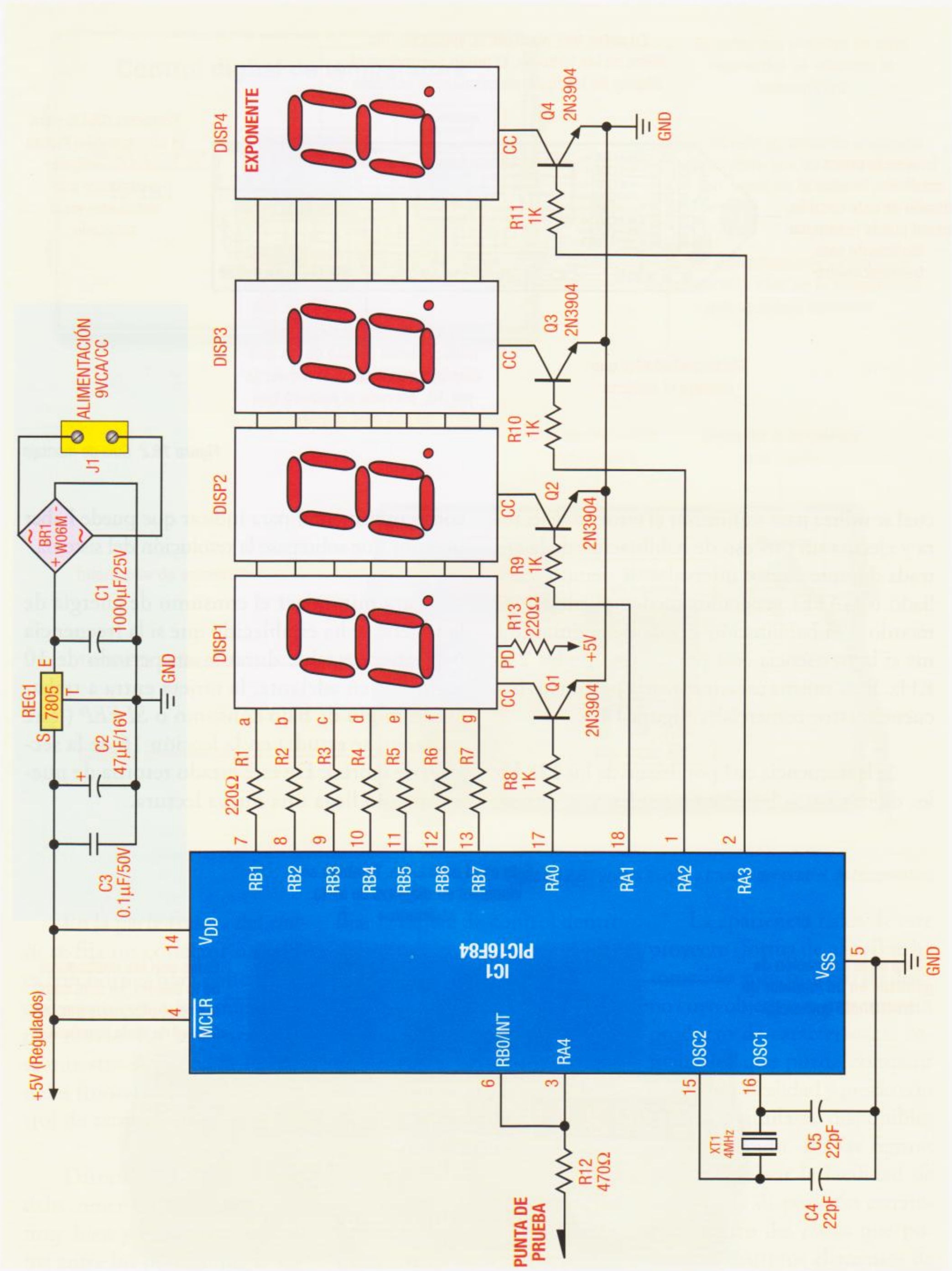


Figura 18.1 *Diagrama esquemático del circuito*

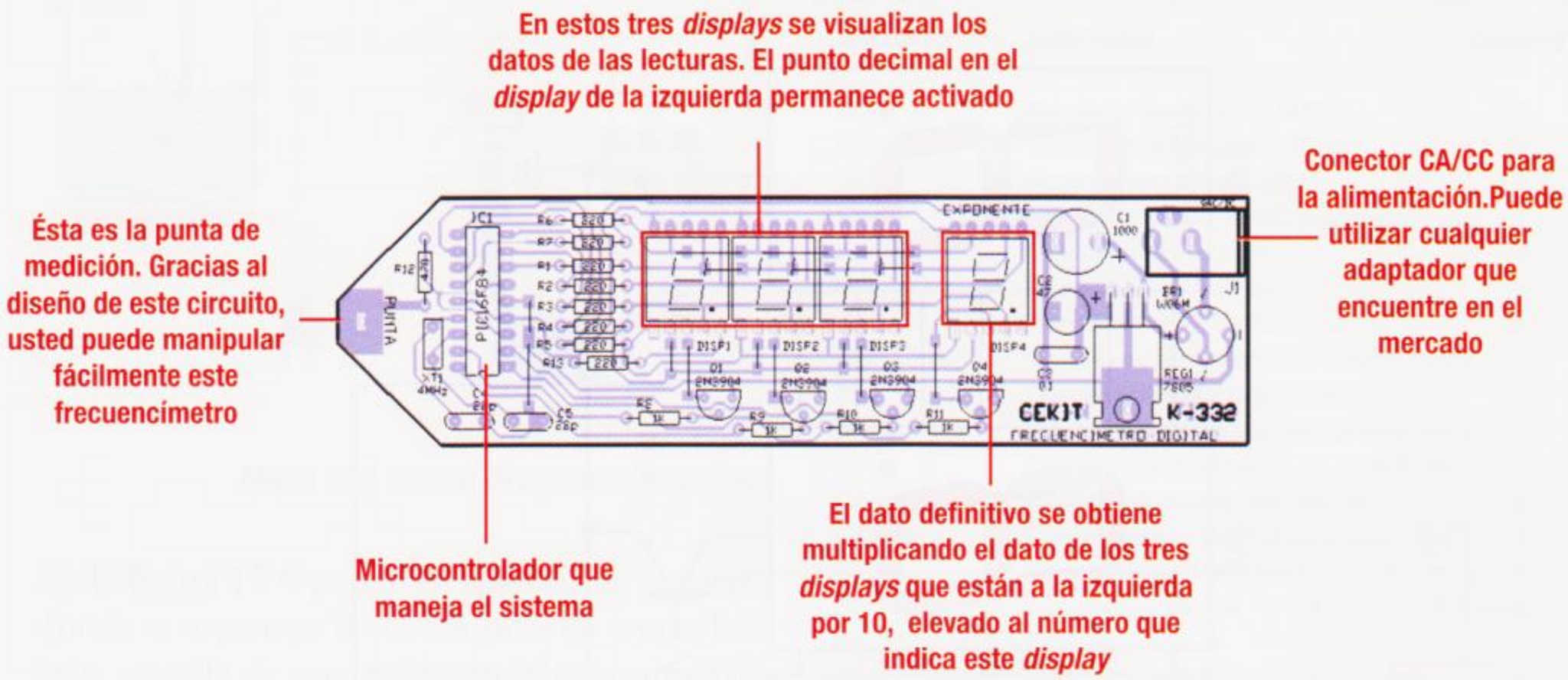


Figura 18.2 *Guía de montaje*

cual se utiliza para minimizar el error en la lectura y ejecuta un proceso de habilitación de la entrada durante ciertos intervalos de tiempo (gatillado ó GATE) generados mediante rutinas de retardo. Esa habilitación se mantiene durante 1 ms si la frecuencia está por encima de los 256 KHz. Esta misma característica la poseen los frecuencímetros comerciales, **figura 18.3**.

Si la frecuencia está por debajo de los 128 Hz los digitos parpadean (se encienden y se apagan consecutivamente) para indicar que puede haber un error que sobrepase la resolución del sistema.

Para minimizar el consumo de energía de la tarjeta, se ha establecido que si la frecuencia permanece estable durante un periodo de 10 segundos en adelante, la tarjeta entra a trabajar en modo de bajo consumo o *SLEEP* (este concepto se estudia en la lección 16 de la sección de teoría). De este estado retorna de nuevo cuando llega una nueva lectura.

Figura 18.3 *Indicación del gatillado, en un frecuencímetro comercial*

Como se estudió en la lección 16 de la sección de teoría, cuando el microcontrolador entra al modo **sleep** muchas de sus funciones se desactivan permitiendo reducir al máximo su consumo de energía. En este caso dicha característica puede ser aprovechada para instalar en la tarjeta un sistema de alimentación por batería y que el instrumento pueda ser portátil.

Descripción del programa

Por razones de espacio se hace imposible publicar el código completo del programa del frecuencímetro digital; en este espacio explicaremos las principales rutinas y su desarrollo. El código completo de este proyecto lo puede descargar de nuestra página en la internet ***www.cekit.com.co*** para que lo utilice libremente.

En la **figura 18.4** podemos apreciar la primera parte del programa, que corresponde a la definición del mapa de memoria RAM y los registros de este proyecto; aquí se han definido direcciones para cada una de las variables del proyecto, para cada uno de los *displays*, los distintos contadores, etc.

El cuerpo principal del programa debe seguir la rutina mostrada en el diagrama de flujo de la **figura 18.5**, en ella se indica que se debe hacer una configuración inicial de los registros OPCION e INTCON de tal manera que se establezca que se van a leer los pulsos de entrada por el pin TOCK y que esos pulsos van a ser la fuente de señal para el incremento del Timer0 (TMR0); de esta manera se cuentan los pulsos de entrada. Los pulsos que entran al sistema los tiene disponibles dentro de este registro y ejecutando operaciones sobre el contenido que éste vaya tomando en el tiempo, puede calcular un dato equivalente a la frecuencia.

Los registros de configuración se muestran en la **figura 18.6** y la parte del programa que ejecuta esta operación se muestra en la **figura 18.7**. Para comprobar el correcto funcionamiento de los *displays*, el pro-

```
;--------------------
;CURSO PRÁCTICO SOBRE MICROCONTROLADORES
; PEREIRA 2001 - AUTOR: JACW
; FASCICULOS 20 Y 21
;--------------------
        LIST    P=16F84

PC      EQU     02      ;CONTADOR DE PROGRAMA
PORTA   EQU     05
PORTB   EQU     06
ESTADO  EQU     03
TMR     EQU     01
TRISA   EQU     0X5
TRISB   EQU     0X6
INTCON  EQU     0X0B        ;REGISTRO DE CONTROL DE INTERRUPCIONES
UNIDAD  EQU     0X0C        ;DISPLAY UNIDADES
DECIMAL EQU     0X1D        ;DISPLAY DECENAS
CENTS   EQU     0X1E        ;DISPLAY CENTENAS
EXP     EQU     0X1F        ;DISPLAY EXPONENTE
BYTE_AL EQU     0X10        ;BYTE ALTO DEL NUMERO LEÍDO (N)
BYTE_BA EQU     0X11        ;BYTE ALTO DEL NUMERO LEÍDO(N)
CONTA_R EQU     0X12        ;ACTUALIZAR CONTADOR
CIFRA   EQU     0X13        ;PARÁMETRO
U       EQU     0X14        ;UNIDADES
D       EQU     0X15        ;DECENAS
H       EQU     0X16        ;CENTENAS
M       EQU     0X17        ;MILES
DM      EQU     0X18        ;DECENAS DE MILES
CM      EQU     0X19        ;CENTENAS DE MILES
CONTA1  EQU     0X1A        ;PRIMER REGISTRO DE CUENTA
CONTA2  EQU     0X1B        ;SEGUNDO REGISTRO DE CUENTA
N       EQU     0X1C        ;
CERO    EQU     0X20        ;NUMERO DE CICLOS SIN SE-AL
```

Figura 18.4 *Definición de las direcciones en la RAM*

Figura 18.5 *Diagrama de flujo del cuerpo principal del programa*

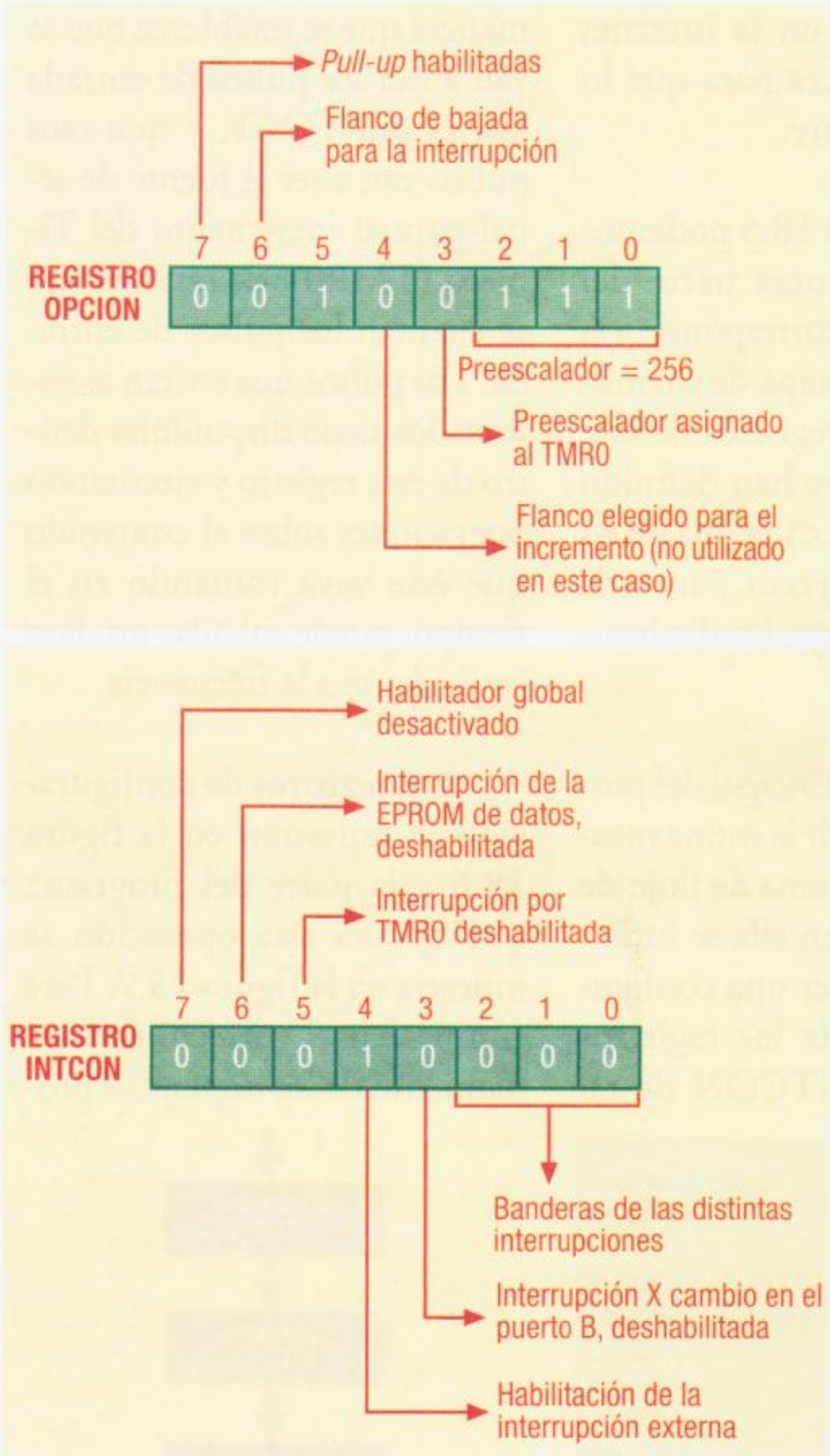


Figura 18.6 *Configuración de los registros OPCION e INTCON*

```
;**********  PRINCIPAL
*************************************
       ORG     0
START
        MOVLW   0X27
        OPTION                  ;CARGAR 00100111 EN
                                ;OPCIÓN
        MOVLW 0X10
        MOVWF   INTCON          ;HABILITAR
                                ;INTERRUPCIONES
        MOVLW .20
        MOVWF   CERO
        CLRF    PORTA
        CLRF    PORTB
        BSF     ESTADO,5
        MOVLW   0X10
        MOVWF   TRISA           ;BITS 0 A 3 DEL
                                ;PUERTO A COMO SALIDAS
                                ;PORTA<4>=ENTRADA
        CLRF    TRISB           ;PUERTO B ->SALIDAS
        BCF     ESTADO,5
        MOVLW   0X08            ;ENCENDER TODOS LOS
                                ;SEGMENTOS PARA
        MOVWF   UNIDAD          ;PROBAR LOS DISPLAYS
        MOVWF   DECIMAL         ;PARA ELLO SE CARGAN CON
                                ;EL NÚMERO
        MOVWF   CENTS           ;OCHO, PARA ENCENDER
                                ;TODOS LOS DISPLAYS
        MOVWF   EXP
        MOVLW   .20             ;ACTUALIZACION DE
                                ;PRUEBA (0,5SEG)
        MOVWF   CONTA_R
LOOP1   CALL    ACTUALIZAR      ;ACTUALIZAR DISPLAYS
        DECFSZ  CONTA_R
        GOTO    LOOP1
LOOP5   CLRF    TMR             ;INICIALIZAR EL
                                ;TMR0 DESDE 0000
        BSF     ESTADO,5        ;CONFIGURAR RB0 COMO
                                ;ENTRADA
        MOVLW   0X01            ;PARA PODER EJECUTAR
                                ;LA INT EXTERNA
        MOVWF   TRISB           ;
        BCF     ESTADO,5        ;RETARDO DE 1 ms
        MOVLW   .99
        MOVWF   CONTA1
```

Figura 18.7 *Código del cuerpo principal*

grama hace una pequeña prueba consistente en encender por un corto espacio de tiempo todos los segmentos de todos los *displays*.

Ahora debe poner en funcionamiento el TMR0, pero solamente por un determinado espacio de tiempo para así tener una medida de los pulsos que entran en un lapso de tiempo. Para lograr esto basta configurar el pin RB0, que es por donde entran los pulsos al registro TMR0, como entrada e invocar un retardo por programa, **figura 18.8** y una vez terminado éste, configurar el pin RB0 como salida para inhibir la entrada de más pulsos. Cuando ha terminado el retardo se envía un flanco por el pin RB0 para originar una interrupción por el pin INT. Si observa el diagrama esquemático de la figura 18.1, puede notar que estos dos pines se encuentran conectados. Para ejecutar de manera continua la lectura de la frecuencia debe repetir el mismo proceso: borrar el TMR0 y habilitar su incremento sólo durante un espacio de tiempo, para poder tomar ese dato y someterlo a un procedimiento donde se aproxime la frecuencia leída.

La lectura del TMR0 se hace dentro de la subrutina PRESCALADOR, **figura 18.9**. En ella se envía el flanco para el pin INT y se establece el valor leído dentro de dos registros denominados BYTE_AL y

```
UN_MS   NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        DECFSZ  CONTA1
        GOTO    UN_MS
        NOP
        NOP
        NOP
        NOP
        NOP
        BSF     ESTADO,5
        CLRF    TRISB
        BCF     ESTADO,5
        CALL    PRESCALADOR         ;UBICAR EL NÚMERO CONTADO EN LOS
                                    ;BYTES ALTO Y BAJO
        MOVF    BYTE_AL
        BTFSS   ESTADO,2            ;TEST: BYTE ALTO=0
        GOTO    CAL1_MS             ;NO
        BTFSC   BYTE_BA,7           ;SI, -> PRUEBE SI BYTE BAJO<=128
        GOTO    CAL1_MS             ;NO
        CLRF    TMR                 ;SI
        BSF     ESTADO,5            ;CONFIGURAR RB0 COMO ENTRADA
        MOVLW   0X01
        MOVWF   TRISB
        BCF     ESTADO,5
        MOVLW   .20
        MOVWF   CONTA_R
```

Figura 18.8 *Retardo por programa e invocación de la rutina para leer el dato adquirido*

```
; *** SUBRUTINA PARA EXTRAER EL VALOR CONTENIDO EN EL PRESCALADOR
PRESCALADOR
        MOVF    TMR,0
        MOVWF   BYTE_AL             ;HACER UNA COPIA DEL CONTADOR EN EL
                                    ;BYTE ALTO
        CLRF    N
CICLO   BCF     PORTB,0
        BSF     PORTB,0
        BCF     PORTB,0             ;DANDO UN FLANCO PARA LA ENTRADA DEL
                                    ;CONTROLADOR
        INCF    N
        MOVF    BYTE_AL,0           ;HACER UNA COPIA DE BYTE ALTO EN W
        XORWF   TMR,0               ;CONTROL SI HA CAMBIADO EL TMR
                                    ;(TMR=BYTE ALTO)
        BTFSC   ESTADO,2
        GOTO    CICLO
        MOVLW   0XFF
        MOVWF   BYTE_BA
        MOVF    N,0
        SUBWF   BYTE_BA             ;BYTE BAJO=256-N
        INCF    BYTE_BA             ;
        RETURN
```

Figura 18.9 *Lectura del TMR0*

BYTE_BAJ. Cuando se tienen listos estos valores se invoca una subrutina en la cual se hace el cálculo exacto de la frecuencia leída, **figura 18.10**, para ello se hace una lectura del valor en los dos registros entregados y según el valor leído se hace un ajuste, sumándole un valor que permita obtener el valor real, calibrado con respecto a un dispositivo comercial para la medida de frecuencia.

Los valores de la frecuencia deben entrar a una subrutina donde se distribuyan entre los cuatro *displays* para su visualización, para ello se han creado variables para las unidades, decenas, centenas y la potencia. Cuando estos datos se han discriminado adecuadamente, se someten al proceso mostrado en la **figura 18.11**, el cual se ejecuta mediante el código mostrado en la **figura 18.12**.

Para llevar a cabo esta operación se hace uso de una variable denominada CIFRA, en la cual se cargan de manera secuencial los valores que se van a visualizar en cada uno de los *displays*. Se invoca una tabla, **figura 18.13**, con los valores configurados para visualizar correctamente los números y se activa cada uno de los transistores que habilitarán el funcionamiento de cada uno de los *displays*, haciendo una especie de barrido. Este concepto lo ha estudiado en proyectos anteriores.

Pruebas de la tarjeta

Uno de los elementos más importantes que se debe tener en cuenta en el momento de desarrollar un proyecto que constituya una herramienta de trabajo, es garantizar la calidad final del aparato; es fundamental que éste haya sido sometido a un conjunto de pruebas que corroboren su correcto funcionamiento y que le garantice estar desarrollando sus trabajos con un elemento confiable que le indique las lecturas con total veracidad.

```
; ************ SUBRUTINA DE CÁLCULO DE LA FRECUENCIA
CÁLCULO MOVLW   .20
        MOVWF   CERO
        RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT7 DEL BYTE ALTO
        GOTO    C_2
        MOVLW   0X03
        ADDWF   DM
        MOVLW   0X02
        ADDWF   M
        MOVLW   0X07
        ADDWF   H
        MOVLW   0X06
        ADDWF   D
        MOVLW   0X08
        ADDWF   U               ;SI BIT7=1 -> SUMAR 32768
C_2     RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT6 DEL BYTE ALTO
        GOTO    C_3
        MOVLW   0X01
        ADDWF   DM
        MOVLW   0X06
        ADDWF   M
        MOVLW   0X03
        ADDWF   H
        MOVLW   0X08
        ADDWF   D
        MOVLW   0X04
        ADDWF   U               ;SI BIT6=1 -> SUMAR 16384
C_3     RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT5 DEL BYTE ALTO
        GOTO    C_4
        MOVLW   0X08
        ADDWF   M
        MOVLW   0X01
        ADDWF   H
        MOVLW   0X09
        ADDWF   D
        MOVLW   0X02
        ADDWF   U               ;SI BIT5=1 -> SUMAR 8192
C_4     RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT4 DEL BYTE ALTO
        GOTO    C_5
        MOVLW   0X04
        ADDWF   M
        MOVLW   0X09
        ADDWF   D
        MOVLW   0X06
        ADDWF   U               ;SI BIT4=1 -> SUMAR 4096
C_5     RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT3 DEL BYTE ALTO
        GOTO    C_6
        MOVLW   0X02
        ADDWF   M
        MOVLW   0X04
        ADDWF   D
        MOVLW   0X08
        ADDWF   U               ;SI BIT3=1 -> SUMAR 2048
C_6     RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT2 DEL BYTE ALTO
        GOTO    C_7
        MOVLW   0X01
        ADDWF   M
        MOVLW   0X02
        ADDWF   D
        MOVLW   0X04
        ADDWF   U               ;SI BIT2=1-> SUMAR 1024
C_7     RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT1 DEL BYTE ALTO
        GOTO    C_8
        MOVLW   0X05
        ADDWF   H
        MOVLW   0X01
        ADDWF   D
        MOVLW   0X02
        ADDWF   U               ;SI BIT1=1 -> SUMAR 512
C_8     RLF     BYTE_AL
        BTFSS   ESTADO,0        ;CHEQUEAR BIT0 DEL BYTE ALTO
        GOTO    C_9
        MOVLW   0X02
        ADDWF   H
        MOVLW   0X05
        ADDWF   D
        MOVLW   0X06
        ADDWF   U               ;SI BIT0=1 ->SUMAR 256
C_9     RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUER BIT7 DE BAJO BYTE
        GOTO    C_10
        MOVLW   0X01
        ADDWF   H
        MOVLW   0X02
        ADDWF   D
        MOVLW   0X08
        ADDWF   U               ;SI BIT7=1 -> SUMAR 128
C_10    RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUEAR BIT6 DEL BYTE BAJO
        GOTO    C_11
        MOVLW   0X06
        ADDWF   D
        MOVLW   0X04
        ADDWF   U               ;SI BIT6=1 -> SUMAR 64
C_11    RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUEAR BIT5 DEL BYTE BAJO
        GOTO    C_12
        MOVLW   0X03
        ADDWF   D
        MOVLW   0X02
        ADDWF   U               ;SI BIT5=1 -> SUMAR  32
C_12    RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUEAR BIT4 DEL BYTE BAJO
        GOTO    C_13
        MOVLW   0X01
        ADDWF   D
        MOVLW   0X06
        ADDWF   U               ;SI BIT4=1 -> SUMAR  16
C_13    RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUEAR BIT3 DEL BYTE BAJO
        GOTO    C_14
        MOVLW   0X08
        ADDWF   U               ;SI BIT3=1 -> SUMAR  8
C_14    RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUEAR BIT2 DEL BYTE BAJO
        GOTO    C_15
        MOVLW   0X04
        ADDWF   U               ;SI BIT2=1 -> SUMAR  4
C_15    RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUEAR BIT1 DEL BYTE BAJO
        GOTO    C_16
        MOVLW   0X02
        ADDWF   U               ;SI BIT1=1 -> SUMAR  2
C_16    RLF     BYTE_BA
        BTFSS   ESTADO,0        ;CHEQUEAR BIT0 DEL BYTE BAJO
        GOTO    C_17
        MOVLW   0X01
        ADDWF   U               ;SI BIT0=1 -> SUMAR  1
C_17    CALL    REPORTE
        MOVF    CM,0            ;EL PRIMER DÍGITO (MSD) SE
BUSCA
        BTFSS   ESTADO,2        ;Y SE HACE UNA APROXIMACIÓN
                                ;AL CUARTO DIGITO
        GOTO    APPCMNZ
        MOVF    DM,0
        BTFSS   ESTADO,2
        GOTO    APPDMNZ
        MOVF    M,0
        BTFSC   ESTADO,2
        GOTO    CUARTO
        MOVF    U,0
        MOVWF   N
        MOVLW   0X05
        SUBWF   N
        BTFSS   ESTADO,0
        GOTO    CUARTO
        INCF    D
        GOTO    CUARTO
APPDMNZ MOVF    D,0             ; DM ES EL MSD
        MOVWF   N
        MOVLW   0X05
        SUBWF   N
        BTFSS   ESTADO,0
        GOTO    CUARTO
        INCF    H
        GOTO    CUARTO
APPCMNZ MOVF    H,0             ;CM ES EL MSD
        MOVWF   N
        MOVLW   0X05
        SUBWF   N
        BTFSS   ESTADO,0
        GOTO    CUARTO
        INCF    M
CUARTO  CALL    REPORTE
        MOVF    CM,0            ;COLOCAR EL 4 DÍGITO EN EL REGISTRO
        BTFSS   ESTADO,2        ;DEL CUARTO DISPLAY
        GOTO    CMNZ            ;Y EN EXP EL VALOR ADECUADO
        .
        .
        .
        .
        .
```

Figura 18.10 *Cálculo de la frecuencia*

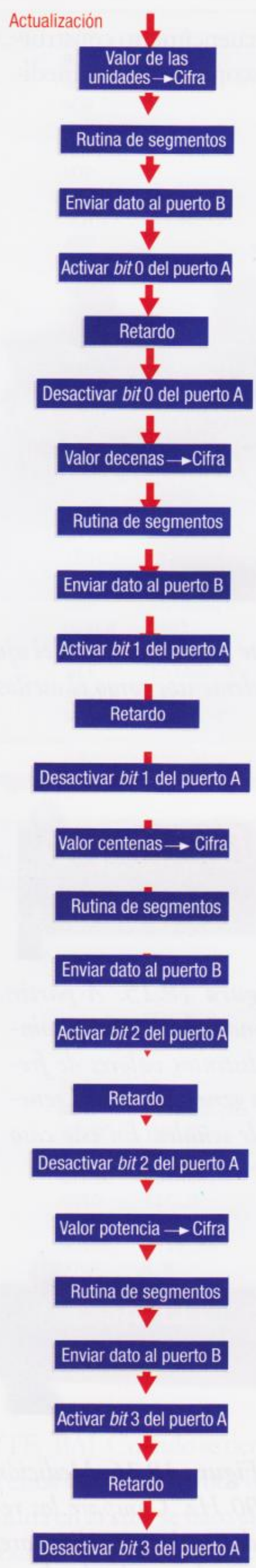


Figura 18.11 *Diagrama de la visualización en los displays*

```
; **** SUBROUTINA DE ACTUALIZACIÓN:  250 MSEC
ACTUALIZAR
        MOVF    UNIDAD,0      ; CARGAR EN EL REGISTRO CIFRA
        MOVWF   CIFRA         ; AL VALOR ACTUALIZADO DE LA VARIABLE
        CALL    SEGMENTOS     ; DE LAS UNIDADES E INVOCAR LA SUBRUTINA
        MOVWF  PORTB          ; QUE CONFIGURA EL DATO QUE SE VA A ENVIAR
        BSF     PORTA,0       ; POR EL PUERTO B, ACTIVAR DISPLAY DE LAS
                              ; UNIDADES
        CALL    RETARDO       ; Y SE ESPERA UN CORTO TIEMPO PARA SU
                              ; VISUALIZACION
        BCF     PORTA,0       ; FINALMENTE SE DESACTIVA EL DISPLAY
        MOVF    DECIMAL,0     ; CARGAR EN EL REGISTRO CIFRA
        MOVWF   CIFRA         ; AL VALOR ACTUALIZADO DE LA VARIABLE
        CALL    SEGMENTOS     ; DE LAS DECENAS E INVOCAR LA SUBRUTINA
        MOVWF  PORTB          ; QUE CONFIGURA EL DATO QUE SE VA A ENVIAR
        BSF     PORTA,1       ; ACTIVAR DISPAY DE LAS DECENAS
        CALL    RETARDO       ; Y SE ESPERA UN CORTO TIEMPO PARA SU
                              ; VISUALIZACIÓN
        BCF     PORTA,1       ; FINALMENTE SE DESACTIVA EL DISPLAY
        MOVF    CENTS,0       ; CARGAR EN EL REGISTRO CIFRA
        MOVWF   CIFRA         ; AL VALOR ACTUALIZADO DE LA VARIABLE
        CALL    SEGMENTOS     ; DE LAS CENTENAS E INVOCAR LA SUBRUTINA
        MOVWF  PORTB          ; QUE CONFIGURA EL DATO QUE SE VA A ENVIAR
        BSF     PORTA,2       ; ACTIVAR DISPLAY DE LAS CENTENAS
        CALL    RETARDO       ; Y SE ESPERA UN CORTO TIEMPO PARA SU
                              ; VISUALIZACIÓN
        BCF     PORTA,2       ; FINALMENTE SE DESACTIVA EL DISPLAY
        MOVF    EXP,0         ; CARGAR EN EL REGISTRO CIFRA
        MOVWF   CIFRA         ; AL VALOR ACTUALIZADO DE LA VARIABLE
        CALL    SEGMENTOS     ; DE LA POTENCIA E INVOCAR LA SUBRUTINA
        MOVWF  PORTB          ; QUE CONFIGURA EL DATO QUE SE VA A ENVIAR
        BSF     PORTA,3       ; ACTIVAR DISPLAY DE LA POTENCIA
        CALL    RETARDO       ; Y SE ESPERA UN CORTO TIEMPO PARA SU
                              ; VISUALIZACIÓN
        BCF     PORTA,3       ; FINALMENTE SE DESACTIVA EL DISPLAY
        RETURN
```

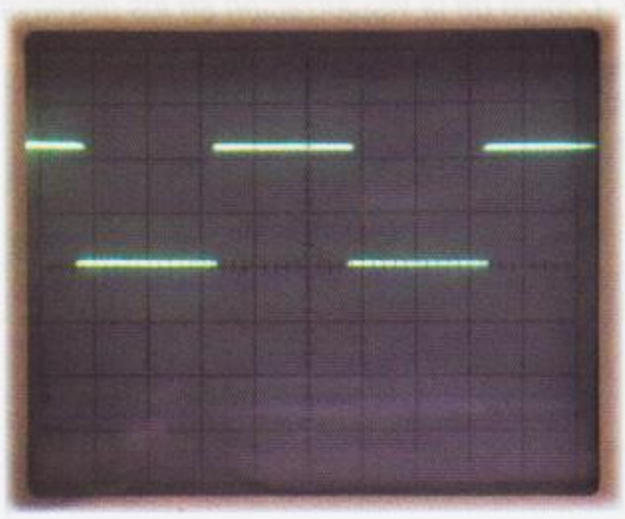
Figura 18.12 *Código de la visualización en los displays*

```
; ****************************    SUBRUTINA DE RETARDO
RETARDO MOVLW   .8                  ; 62.5 ms
        MOVWF   CONTA1
        NOP
BETA    MOVLW   .0
        MOVWF   CONTA2
ALFA    DECFSZ  CONTA2
        GOTO    ALFA
        NOP
        DECFSZ  CONTA1
        GOTO    BETA
        MOVLW   .14
        MOVWF   CONTA2
GAMMA   DECFSZ  CONTA2
        GOTO    GAMMA
        NOP
        NOP
        RETURN
; ****** SELECCIONAR LOS SEGMENTOS ADECUADOS*************
SEGMENTOS
        NOP
        MOVF    CIFRA,0
        ADDWF   PC                  ;RETORNMAR EN W EL CONJUNTO ADECUADO DE
        RETLW   0X7E                ;SEGMENTOS QUE SE ACTIVAN PARA CADA SEGMENTO
        RETLW   0X0C
        RETLW   0XB6
        RETLW   0X9E
        RETLW   0XCC
        RETLW   0XDA
        RETLW   0XFA
        RETLW   0X0E
        RETLW   0XFE
        RETLW   0XDE
```

Figura 18.13 *Tabla de los segmentos para visualizar los números*

Ahora mostramos algunas pruebas de mediciones que se hicieron con el frecuencímetro construido en esta ocasión, comprobadas mediante la comparación con un osciloscopio y con un medidor digital de frecuencia comercial, **figuras 18.14 a 18.20.**

Figura 18.14. *Este es el procedimiento de calibración del dispositivo, parte fundamental en el ajuste de los parámetros y cálculos del programa. Para esta calibración se utilizaron elementos como el osciloscopio, un generador de señales y un medidor de frecuencia comercial.*



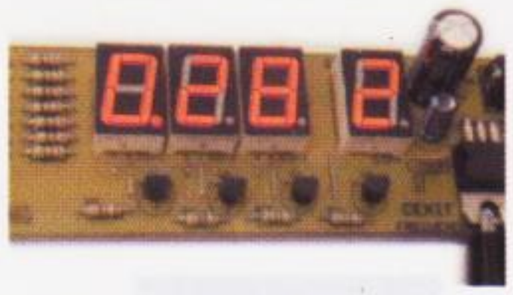


Figura 18.15. *A partir de este momento vamos a comparar distintos valores de frecuencia generados con el generador de señales. En este caso se mide 28 Hz.*

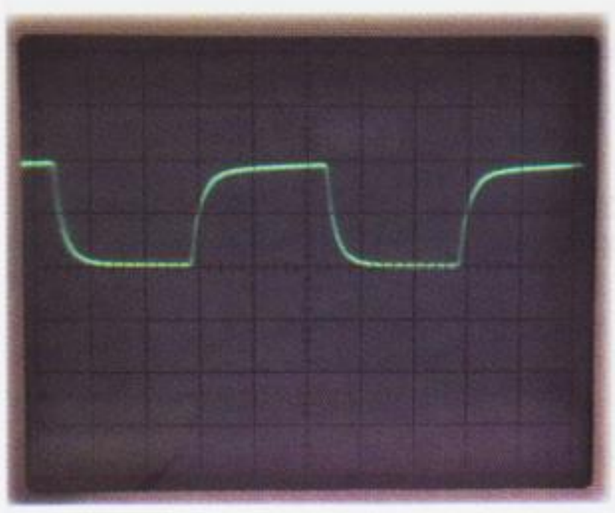


Figura 18.16. *Medición de 100 Hz. Compare los resultados en los visualizadores y acostúmbrese a leerlos de esta manera.*

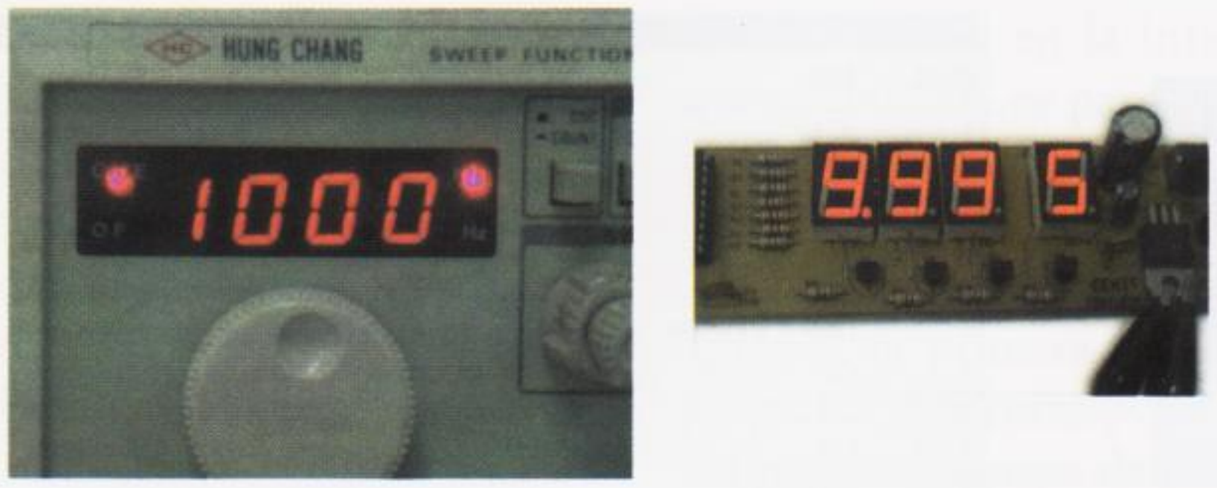


Figura 18.17. Esta es la lectura de 1MHz de frecuencia. La lectura es 9,99 x 10^5.

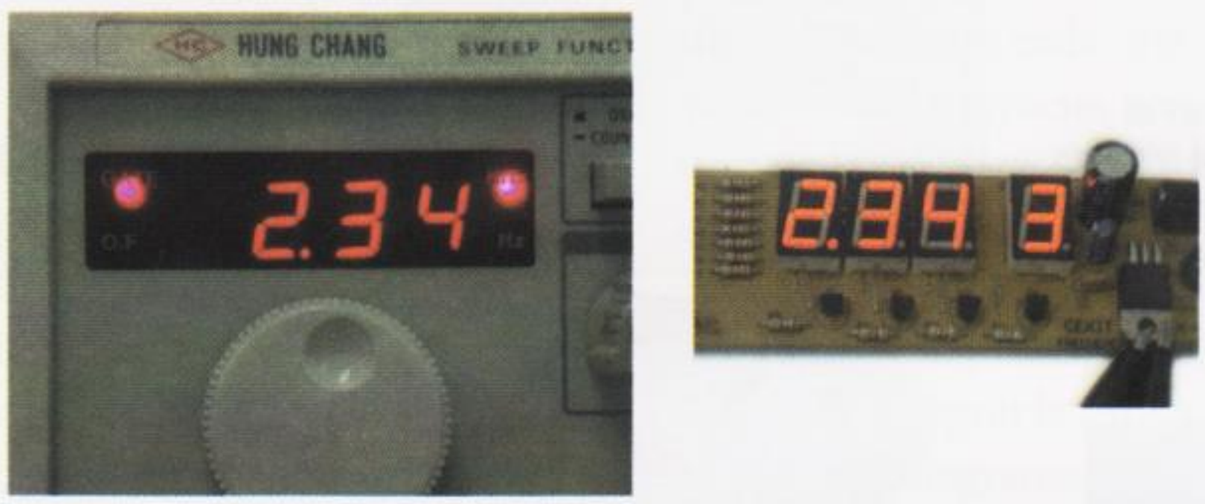


*Figura 18.18. La frecuencia generada es de 2,34 KHz. La lectura en nuestro dispositivo es 2,34 * 10^3.*

*Figura 18.19. En el medidor se puede leer 2.215 KHz, es decir 2,2 MHz, en nuestro dispositivo se lee 2,21 * 10^6.*

Figura 18.20. Debido a la facilidad en su manipulación, el frecuecímetro desarrollado puede ser de gran utilidad para probar toda clase de diseños digitales como relojes, medidores e incluso para hacer pruebas en ciertos circuitos de comunicaciones seriales.

Ensamblaje del proyecto

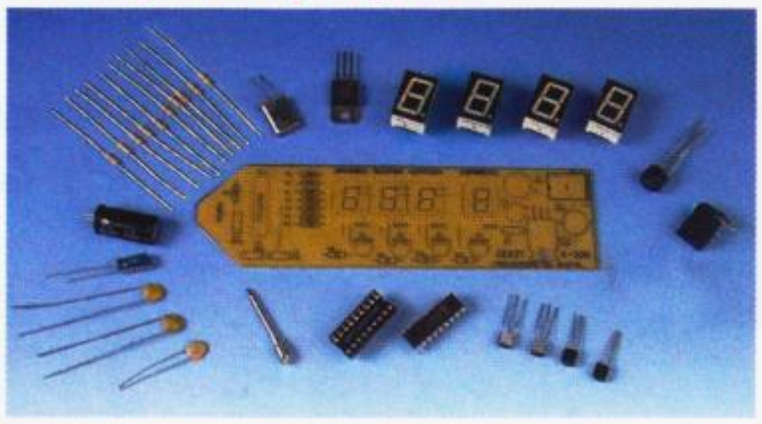


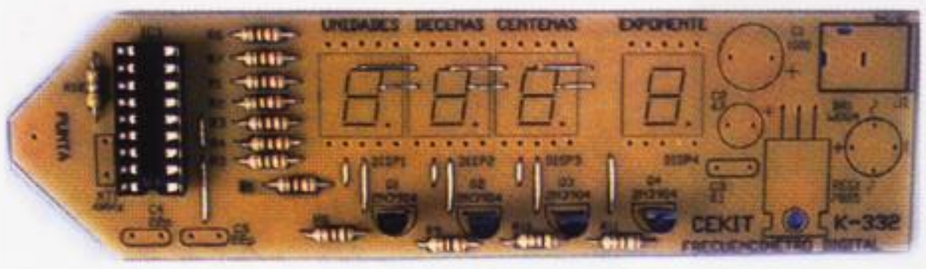
Figura 18.21. *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el frecuencímetro digital CEKIT K-332.*

Una herramienta útil en todo laboratorio de electrónica es el frecuencímetro de señales digitales, tal como el de este proyecto. Con él se puede medir con precisión el valor de la frecuencia de señales entre 0 y 50MHz. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

Paso 1

Instale y suelde los puentes de alambre y las resistencias R1 a R13.

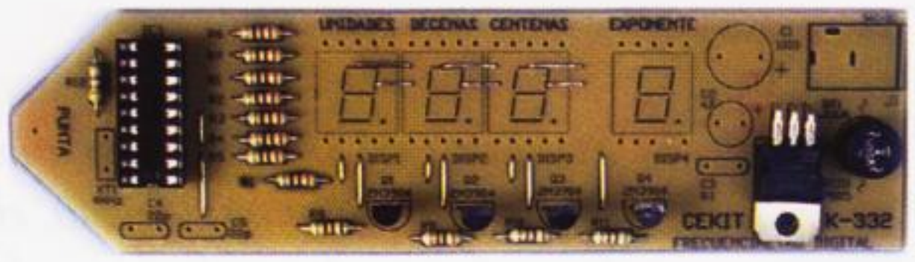
Figura 18.22



Paso 2

Luego instale y suelde los transistores NPN Q1 a Q4 y la base para el microcontrolador.

Figura 18.23

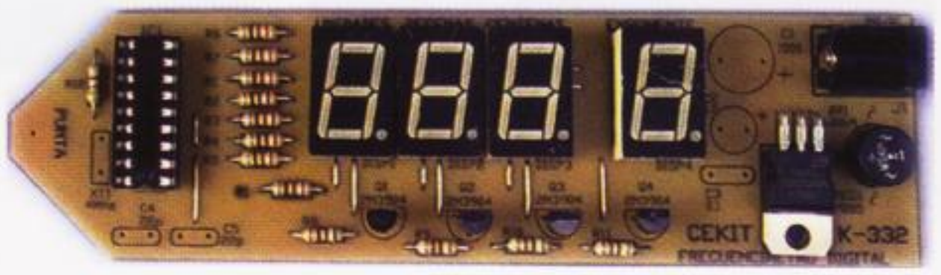


Paso 3

Instale y suelde el puente rectificador BR1 y el regulador de voltaje integrado de +5V REG1.

Figura 18.24

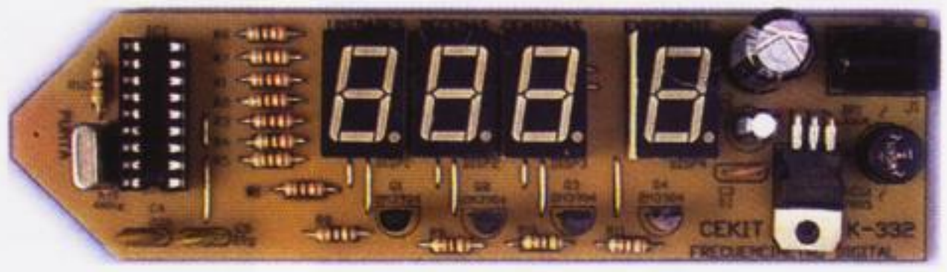
LISTA DE MATERIALES		
Cant.	**Ref.**	**Descripción**
1	IC1	Microcontrolador PIC16F84, programado
1	REG1	Regulador fijo de +5V-1A, 7805
4	Q1 a Q4	Transistores de propósito general NPN, 2N3904
1	BR1	Puente rectificador circular de 1A, W06M
4	DISP1 a DISP4	*Displays* de siete segmentos cátodo común
1	C1	Condensador electrolítico de 1000µF - 25V
1	C2	Condensador electrolítico de 47µF - 25V
1	C3	Condensador cerámico de 0,1µF - 50V
2	C4 y C5	Condensadores cerámicos de 22pF - 50V
1	XT1	Cristal de cuarzo de 4MHz
8	R1 a R7 y R13	Resistencias de 220Ω - 1/4W
4	R8 a R11	Resistencias de 1k - 1/4W
1	R12	Resistencia de 470Ω - 1/4W
1	K-332	Circuito impreso CEKIT
1		Conector de alimentación CA/CC
1		Base para circuito integrado de 18 pines
1		Punta de prueba metálica
1		Lámina de acrílico



Paso 4

Instale y suelde los cuatro displays de cátodo común DISP1 a DISP4 y el conector de alimentación CA/CC J1.

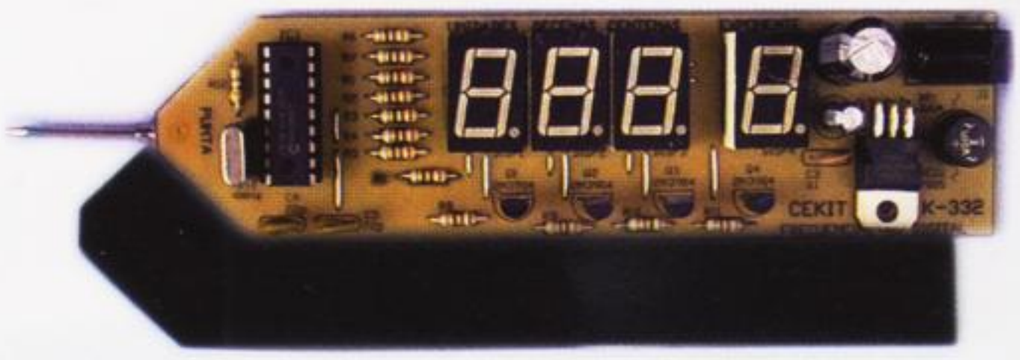
Figura 18.25



Paso 5

Instale y suelde los condensadores cerámicos C3, C4 y C5, el cristal de cuarzo de 4MHz XT1 y los condensadores electrolíticos C1 y C2.

Figura 18.26



Paso 6

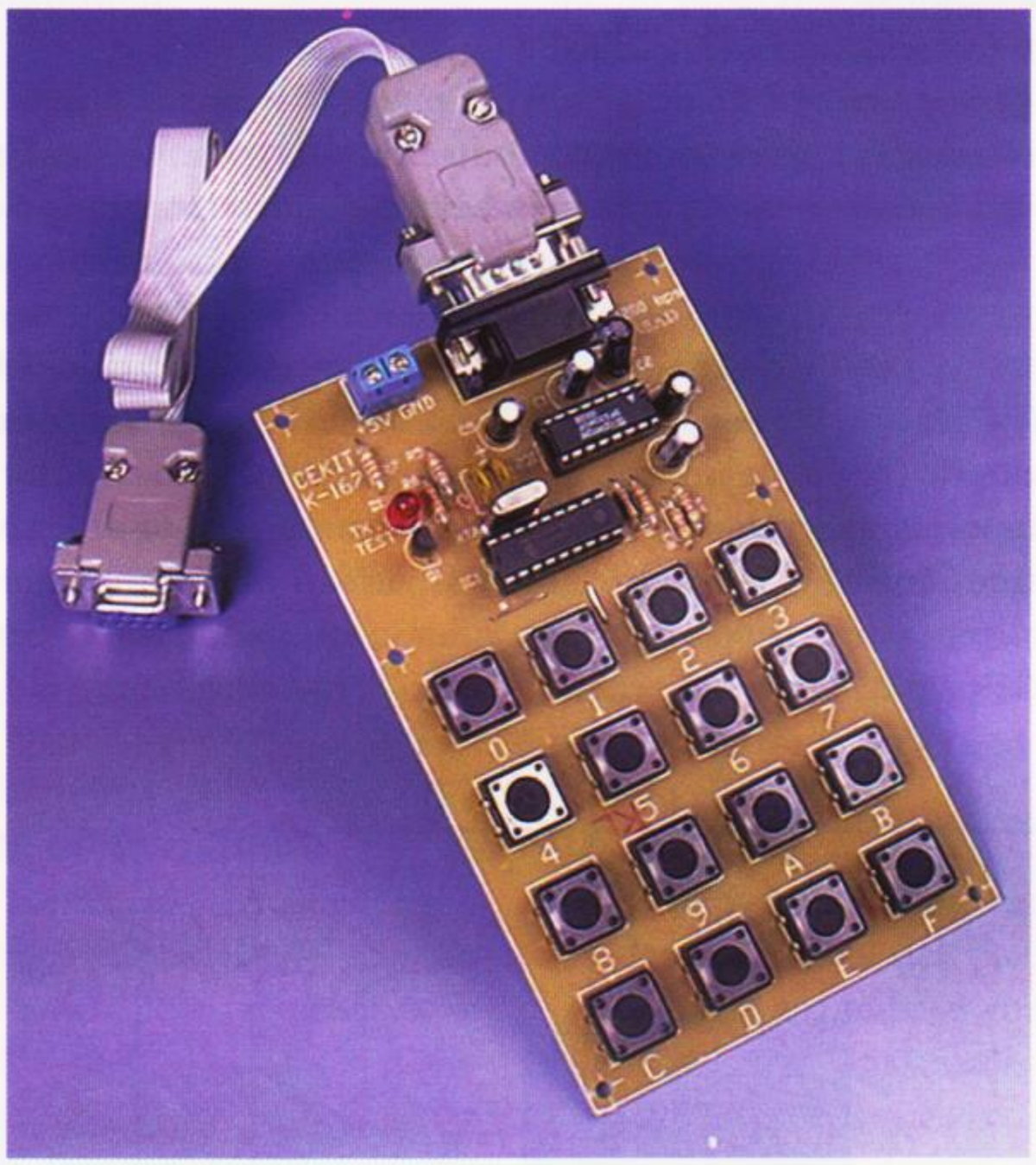
Aspecto final del frecuencímetro luego de adherir con suficiente soldadura la punta de prueba metálica al circuito impreso y en el lugar destinado para dicho componente, marcado en la guía de ensamblaje como PUNTA. Antes de usar el frecuencímetro es recomendable usar una lámina de acrílico para evitar medidas erróneas, pegada por el lado de soldadura del circuito impreso.

Figura 18.27

Paso 6. Pruebas finales

Para alimentar el circuito debe usar un adaptador que brinde un voltaje continuo entre 8 y 12 voltios. Es recomendable instalar una lámina de acrílico sobre la parte inferior de la tarjeta para proteger la capa de soldadura del circuito impreso. Para conectar la tierra de la señal que va a medir utilice la carcasa del regulador 7805.

PROYECTO 19



TECLADO 4X4 CON SALIDA SERIAL

Con este proyecto usted construirá un teclado que puede entregar el valor de la tecla oprimida empleando un solo cable, permitiendo así ahorrarle pines al dispositivo al cual se le entrega el dato.

Características

- Lectura y transmisión serial de los datos
- Visualización de los datos en la pantalla de la computadora
- Velocidad de transmisión de 1.200 bps
- Diodo LED indicador de la transmisión

Como ya lo hemos visto, las comunicaciones seriales son una gran alternativa cuando se trabaja en el diseño y el desarrollo de productos electrónicos. Aparte de las ventajas de transmisión de información a grandes distancias, ellas posibilitan el control de diferentes variables utilizando tan sólo una entrada o salida de un sistema controlador. Este tipo de comunicación está presente en las computadoras personales, en los microcontroladores más especializados, y es muy fácil de implantar con los microcontroladores más sencillos.

En algunos proyectos electrónicos se requiere la utilización de un teclado para que el usuario pueda digitar algún parámetro, un valor de calibración o una clave de seguridad, entre otros. El problema en estos casos es que la conexión de dicho dispositivo ocupa casi todos los pines del microcontrolador, dejando unos pocos para hacer las otras tareas que se deben ejecutar. Por tal razón, en este proyecto usted va a construir un teclado de 16 posiciones, organizado en forma matricial, cuya única función es enviar por su salida, en formato RS-232, el dato correspondiente a la tecla que se oprime.

Aplicaciones

Este dispositivo se puede conectar al puerto serial de una computadora para muchas tareas, por ejemplo en la industria, donde un operario debe reportar los datos correspondientes a algún proceso y las condiciones ambientales no permiten dejar toda la computadora en dicho sitio. Otro ejemplo consiste en ubicar el teclado a la entrada de un área de acceso restringido, en este caso se usaría para que las personas que deseen entrar digiten la clave de seguridad y la computadora o un sistema con microcontrolador pueda determinar si permite o no la entrada.

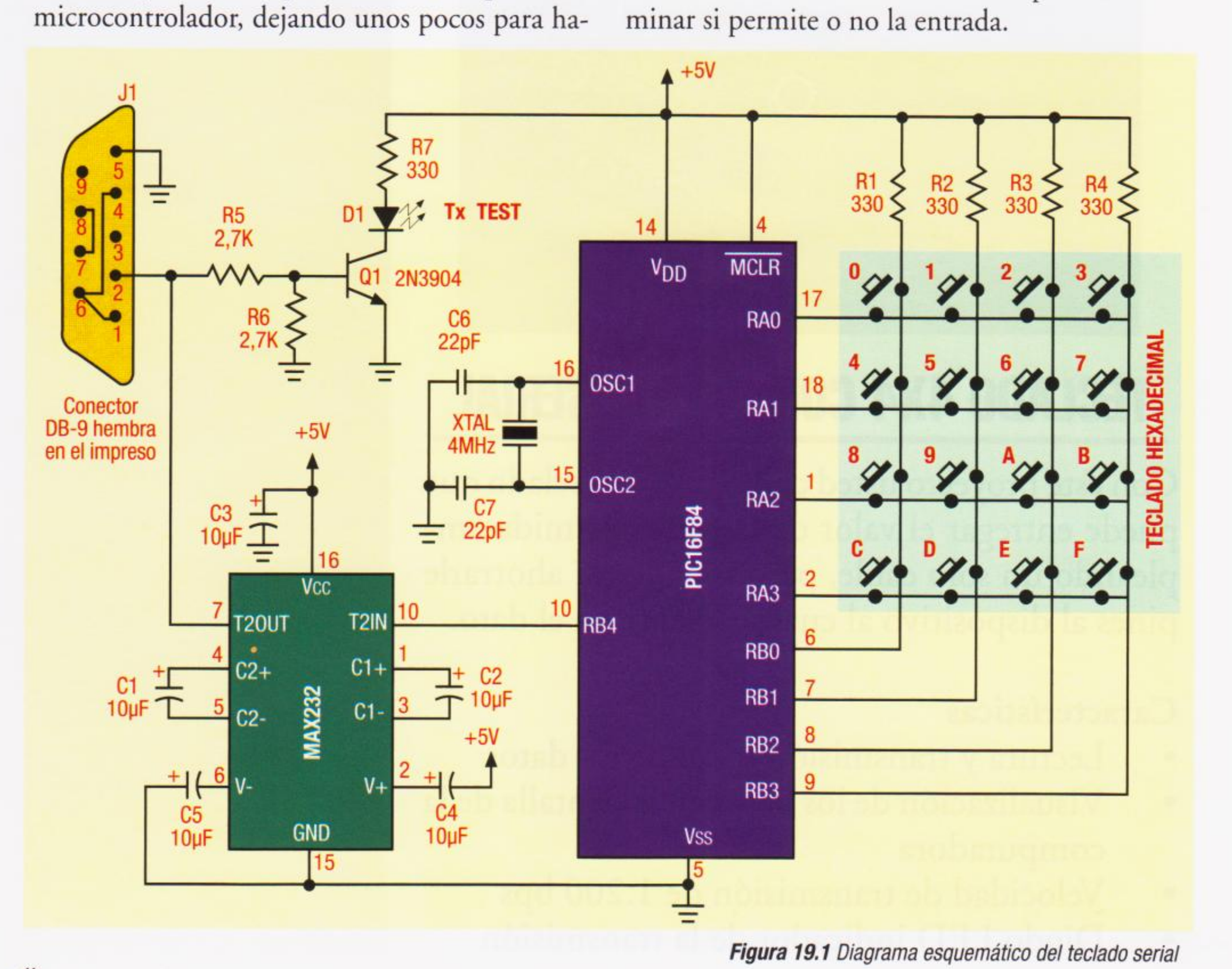


Figura 19.1 Diagrama esquemático del teclado serial

En sistemas microcontrolados la conexión de un teclado emplea una gran cantidad de pines, complicando el diseño de cierta forma ya que los recursos disponibles se ven algo limitados. En estos casos, el uso de este teclado serial permite que queden libres el mayor número posible de pines del microcontrolador ya que, para recibir los datos escritos, sólo se requiere de una entrada.

Funcionamiento

En la **figura 19.1** se muestra el diagrama esquemático del circuito y como se puede ver, el control lo ejerce un microcontrolador PIC16F84. Este elemento se encarga de hacer el barrido de todas las teclas para detectar si alguna de ellas ha sido oprimida; en caso afirmativo, lo que hace es pasar el dato en forma serial, por el pin RB4, hacia el integrado MAX232. Este último tiene como función convertir los niveles lógicos que maneja el PIC16F84 y adaptarlos a los niveles de voltaje de la interfaz RS-232, es decir, un voltaje negativo entre -3 y -15 voltios para representar un uno lógico, y un voltaje positivo entre +3 y +15 voltios para un cero lógico. Para obtener estos niveles de tensión a partir de una fuente simple de 5 voltios, el MAX232 se vale de los condensadores C1 a C5 que tiene conectados.

La salida del MAX232 se lleva directamente hacia el conector DB-9 hembra para circuito impreso. En este conector el pin utilizado para sacar los datos es el número 2, el cual coincide con el pin de recepción del puerto serial COM1 de las computadoras personales. El pin 5 de dicho conector se conecta a tierra y también coincide con el respectivo pin de la computadora, esto se hace con el fin de que el cable de conexión que se utilice, entre el circuito del teclado y la computadora, sea un cable donde los hilos van uno a uno, es decir que no se forman cruces, lo que facilita su construcción y disminuye la posibilidad de errores. En la **figura 19.2** se muestran las conexiones que debe tener el cable que comunica el teclado con una computadora o con algún otro dispositivo al que sea conectado.

El transistor Q1 que recibe en su base la señal de salida del MAX232, se utiliza para activar el LED D1. Este elemento sirve como indicador de la presencia de datos en la salida del circuito; de esta forma se pueden descartar posibles fallas ya que la variación de la intensidad del LED en el momento justo de la transmisión permite establecer un correcto funcionamiento del sistema. Adicionalmente, se debe anotar que el teclado sólo requiere una fuente de alimentación de +5 voltios.

En la **figura 19.3** se muestra la guía de montaje y el circuito impreso del teclado serial; aunque son pocos elementos y de fácil reconocimiento, se recomienda tener especial cuidado sobre todo con la polaridad de los condensadores polarizados que utiliza el integrado MAX232, ya que el símbolo que identifica el pin positivo suele ser muy pequeño. Además, el valor de cada componente se encuentra en la lista de materiales con su respectiva referencia para el montaje sobre la tarjeta.

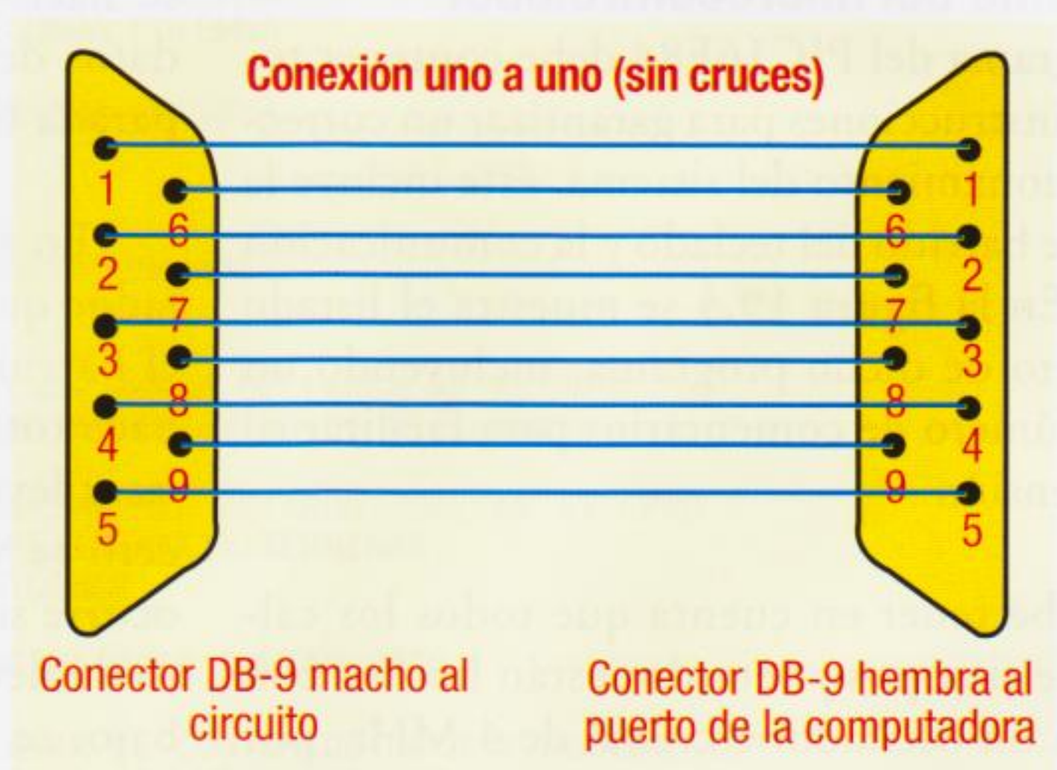


Figura 19.2 *Cable serial para comunicar el teclado con una computadora*

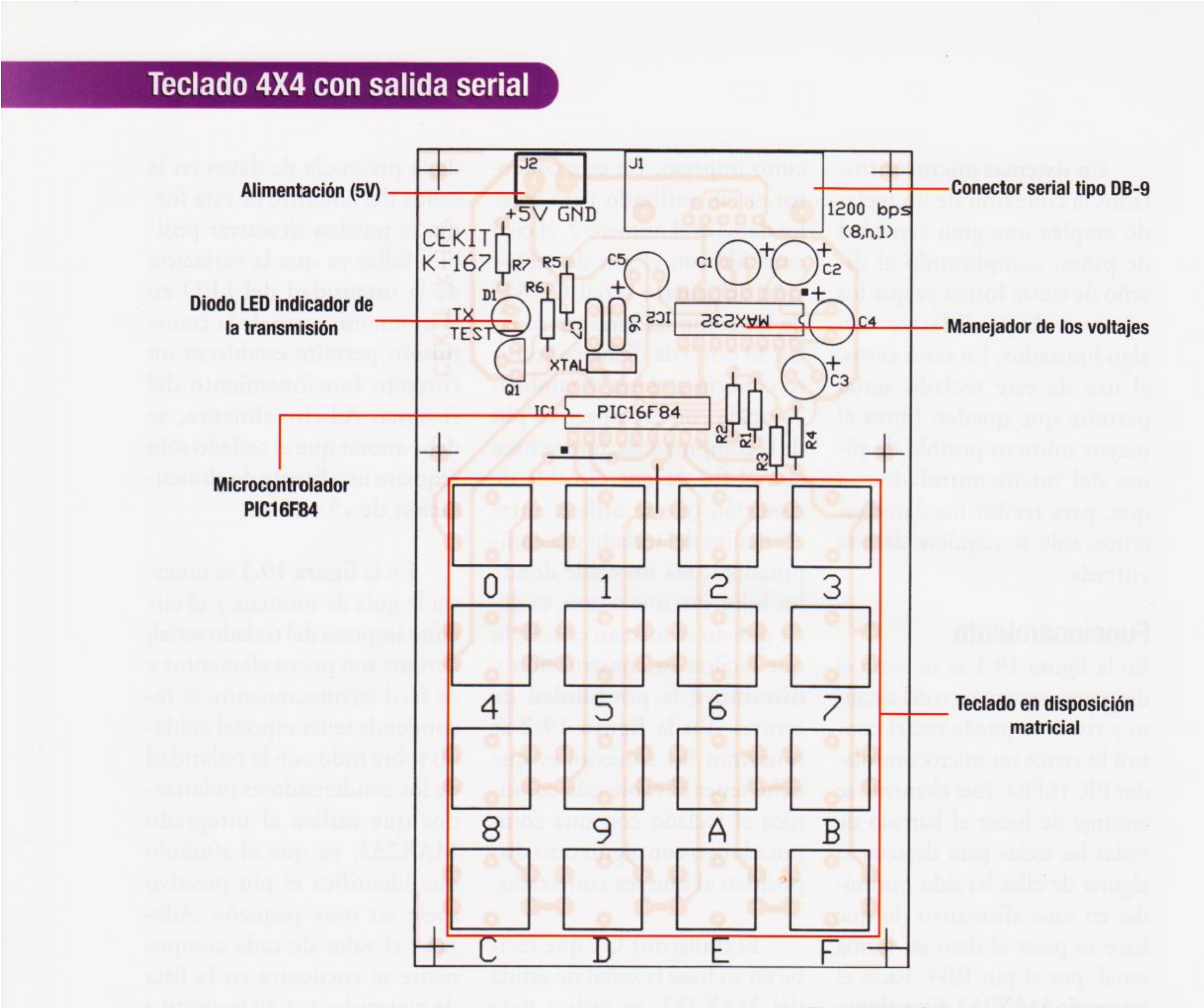


Figura 19.3 Guía de montaje del teclado serial

Programa del microcontrolador

El programa del PIC16F84 debe contener todas las instrucciones para garantizar un correcto funcionamiento del sistema, éste incluye la parte de barrido del teclado y la comunicación serial. En la **figura 19.4** se muestra el listado completo de dicho programa, incluyendo un buen número de comentarios para facilitar su comprensión.

Debe tener en cuenta que todos los cálculos de tiempos y retardos están hechos con base en un oscilador a cristal de 4 MHz, por lo tanto, si desea construir este circuito con otro tipo de oscilador debe ajustar las constantes de tiempo. La transmisión de los datos se hace a una velocidad de 1.200 bps, con datos de 8 *bits*, sin paridad y con un *bit* de parada (*stop bit*).

En el programa principal el microcontrolador queda enclavado en un ciclo que hace el barrido del teclado; este procedimiento se hace rotando un cero por cada una de las filas y leyendo las columnas para ver si dicho cero se ve reflejado en alguna de ellas (esto ocurre si se oprime una tecla). En el momento de detectar la presencia de un nivel lógico bajo, se hace un tratamiento del valor leído para obtener el valor de la tecla que fue oprimida. Después de esto, dicho valor debe ser enviado serialmente hacia el exterior.

```
;ESTE PROGRAMA LEE UN TECLADO MATRICIAL DE 4X4 Y
;ENVIA EL DATO OBTENIDO HACIA UNA COMPUTADORA VIA RS-232

; DEFINICIÓN DE REGISTROS
PC        EQU       02H            ;CONTADOR DE PROGRAMA
STATUS    EQU       03H            ;REGISTRO DE ESTADOS
PTOA      EQU       05H            ;EL PUERTO A ESTÁ  EN LA DIRECCIÓN 05 DE LA RAM
PTOB      EQU       06H            ;EL PUERTO B ESTÁ  EN LA DIRECCIÓN 06 DE LA RAM
TECLA     EQU       0CH            ;CONTIENEN EL VALOR DE LA TECLA OPRIMIDA
LOOPS     EQU       0DH            ;UTILIZADO EN RETARDOS (MILISEGUNDOS)
LOOPS2    EQU       0EH            ;UTILIZADO EN RETARDOS
ROTA      EQU       0FH            ;REGISTRO QUE SE ROTA PARA ENVIAR CEROS A LAS FILAS
FILAS     EQU       10H            ;CONTIENE EL NÚMERO DE LA FILA QUE SE VA A PROBAR
R11       EQU       11H            ;USADO PARA CONTAR BITS TRANSMITIDOS
R12       EQU       12H            ;USADO PARA TIEMPO DE BIT SOBRE LA LÍNEA
TRANS     EQU       13H            ;BYTE QUE SE TRANSMITE
TRISA     EQU       85H            ;REGISTRO DE CONFIGURACIÓN DEL PUERTO A
TRISB     EQU       86H            ;REGISTRO DE CONFIGURACIÓN DEL PUERTO B
Z         EQU       02H            ;BANDERA DE CERO DEL REGISTRO DE ESTADOS
C         EQU       00H            ;BANDERA DE CARRY DEL REGISTRO DE ESTADOS
W         EQU       00H            ;INDICA QUE EL RESULTADO SE GUARDA EN W
TX        EQU       04H            ;PIN DE SALIDA SERIAL DEL PUERTO B

RESET     ORG       0              ;EL VECTOR DE RESET ES LA DIRECCIÓN 00
          GOTO      INICIO         ;SE SALTA AL INICIO DEL PROGRAMA

          ORG       5              ;EL PROGRAMA EMPIEZA EN LA DIRECCIÓN DE MEMORIA 5

RETARDO   ;SUBRUTINA DE RETARDO DE 200 MILISEGUNDOS
          MOVLW     D'200'         ;EL REGISTRO LOOPS CONTIENE EL NÚMERO
          MOVWF     LOOPS          ;DE MILISEGUNDOS DEL RETARDO
TOP2      MOVLW     D'110'         ;
          MOVWF     LOOPS2         ;
TOP       NOP
          NOP
          NOP
          NOP
          CLRWDT
          NOP
          DECFSZ    LOOPS2         ;PREGUNTA SI TERMINO 1 MS
          GOTO      TOP
          DECFSZ    LOOPS          ;PREGUNTA SI TERMINA EL RETARDO
          GOTO      TOP2
          RETLW     0

TABLA     ADDWF     PC             ;SUMAR W AL PC
          NOP
          RETLW     0              ;PRIMERA COLUMNA
          RETLW     1              ;SEGUNDA COLUMNA
          NOP
          RETLW     2              ;TERCERA COLUMNA
          NOP
          NOP
          NOP
          RETLW     3              ;CUARTA COLUMNA

DELAY1    MOVLW     .166           ;RETARDO
          MOVWF     R12            ;LLEVAR VALOR DE CARGA AL RETARDO
REDO      NOP                      ;LIMPIAR CIRCUITO DE VIGILANCIA
          NOP
          DECFSZ    R12            ;DECREMENTAR RETARDO, SALTAR SI CERO
          GOTO      REDO           ;REPETIR HASTA TERMINAR
          RETLW     0              ;RETORNAR

ENVIAR                             ;RUTINA PARA ENVIAR DATO
          CLRWDT
          MOVWF     TRANS          ;LLEVAR EL CONTENIDO DE W A TRANSMISIÓN
XMRT      MOVLW     8              ;CARGAR CON NÚMERO DE BITS
          MOVWF     R11            ;EL CONTADOR
          BCF       PTOB,TX        ;COLOCAR LÍNEA DE TRANSMISIÓN EN BAJO
          CALL      DELAY1         ;PARA GENERAR BIT DE  ARRANQUE
XNEXT     BCF       PTOB,TX        ;COLOCAR LÍNEA DE TRANSMISIÓN EN BAJO
```

```
        BCF     STATUS,C            ;LIMPIAR CARRY
        RRF     TRANS               ;ROTAR REGISTRO DE TRANSMISIÓN
        BTFSC   STATUS,C            ;PREGUNTAR POR EL CARRY
        BSF     PTOB,TX             ;SI ES UNO, COLOCAR LÍNEA EN ALTO
        CALL    DELAY1              ;LLAMAR RETARDO DE 1 BIT
        DECFSZ  R11                 ;DECREMENTAR CONTADOR, SALTAR SI CERO
        GOTO    XNEXT               ;REPETIR HASTA TRANSMITIR TODO EL DATO
        BSF     PTOB,TX             ;COLOCAR LÍNEA DE TRANSMISIÓN EN ALTO
        CALL    DELAY1              ;LLAMAR RETARDO 1 BIT -BIT DE PARADA-
        RETLW   0                   ;RETORNAR

; **** PROGRAMA PRINCIPAL ****

INICIO    BSF       STATUS,5        ;SE UBICA EN EL SEGUNDO BANCO DE RAM
          MOVLW     0F0H            ;SE CARGA EL REGISTRO W CON 0F0H
          MOVWF     TRISA           ;SE PROGRAMAN LOS PINES DEL PUERTO A COMO SALIDAS
          MOVLW     0EFH            ;SE CARGA EL REGISTRO W CON 0EFH
          MOVWF     TRISB           ;SE PROGRAMA EL PUERTO B COMO ENTRADAS Y SALIDAS
          BCF       STATUS,5        ;SE UBICA EN EL PRIMER BANCO DE MEMORIA RAM

          BSF       PTOB,TX         ;INICIA LÍNEA DE SALIDA EN ALTO
          MOVLW     00H             ;PARA EMPEZAR SE INICIA CON UN 0
          MOVWF     TECLA           ;


ESCAN     CLRF      FILAS
          MOVLW     B'1110'         ;SE PREPARA PARA ENVIAR CEROS A LAS FILAS
          MOVWF     ROTA
PROBAR    MOVF      ROTA,W          ;ENVÍA EL DATO A LAS FILAS
          MOVWF     PTOA
          NOP                       ;TIEMPO PARA ESTABILIDAD DE LAS LÍNEAS
LEER      MOVF      PTOB,W          ;LEER LAS COLUMNAS CONECTADAS AL PUERTO B
          ANDLW     0FH             ;ELIMINA LA PARTE ALTA DEL BYTE LEÍDO
          XORLW     0FH             ;INVIERTE EL DATO PARA VER SI HAY ALGÚN CERO
          BTFSS     STATUS,Z        ;PREGUNTA SI EL RESULTADO ES CERO (ALGUNA TECLA)
          GOTO      SALIR           ;SI HAY TECLA VA A TRANSMITIRLA SERIALMENTE
          CLRWDT
          BTFSS     ROTA,3          ;CONSULTA SI YA VAN 4 ROTACIONES
          GOTO      ESCAN           ;SI TERMINÓ, VUELVE A EMPEZAR EL ESCANER DE TECLADO
          BSF       STATUS,C        ;COLOCA BIT DE CARRY EN 1
          RLF       ROTA            ;PARA ROTAR EL 0 QUE VA A IR HACIA LAS FILAS
          MOVLW     4               ;CARGA W CON 4 PARA SUMARLO AL VALOR DE FILAS
          ADDWF     FILAS,1
          GOTO      PROBAR          ;VA A HACER LA PRÓXIMA PRUEBA CON EL 0 ROTADO

SALIR     CALL      TABLA           ;PARA OBTENER VALOR DE LA COLUMNA
          ADDWF     FILAS,W         ;SUMAR COLUMNA Y FILAS PARA OBTENER EL DATO REAL
          MOVWF     TECLA           ;GUARDA EL DATO EN EL REGISTRO TECLA

          SUBLW     09H             ;PRUEBA SI ES IGUAL O MENOR QUE 9
          MOVF      TECLA,W
          BTFSS     STATUS,C        ;SI ESTÁ  ENRTE 0 Y 9 SE LE SUMA 48
          GOTO      MAYOR           ;SI ES MAYOR A 9 SE LE SUMA 54
          ADDLW     D'48'           ;SE LE SUMA ESTE VALOR A TECLA PARA
          GOTO      MENOR           ;OBTENER SU EQUIVALENTE ASCII
MAYOR     ADDLW     D'55'
MENOR     CALL      ENVIAR          ;ENVÍA SERIALMENTE EL VALOR ASCII DE TECLA
          CALL      RETARDO         ;RETARDO ESPERANDO QUE LIBEREN LA TECLA
          GOTO      ESCAN           ;VUELVE A PROBAR SI OPRIMEN ALGUNA TECLA
          END

;=================================================================
;         FUSIBLES DE PROGRAMACIÓN

;         OSC                 XT
;         WATCHDOG            ON
;         CODE PROTECT        OFF
;         POWER-UP-TIMER      ON
;         MICRO.              PIC16F84
;=================================================================
```

Figura 19.4 *Programa del microcontrolador PIC del teclado serial*

La subrutina llamada *enviar* recibe un dato en el registro W y se encarga de todo el procedimiento de desplazamiento de *bits* y generación de retardos para llevar a cabo la transmisión.

Antes de pasar el dato a la rutina de transmisión, se verifica si el dato es un número del teclado entre 0 y 9 ó entre A y F, esto con el fin de sumarle el valor adecuado para convertir el número en su equivalente ASCII. La razón es que de esta forma el dato llega a su destino, por ejemplo la computadora, y puede ser pasado directamente a la pantalla.

Programa de prueba con una computadora

La forma más sencilla de comprobar que el teclado está funcionando correctamente es recibiendo los datos en una computadora. Para ello, conecte el teclado en uno de los puertos seriales de la misma, utilizando el cable que se describió antes. El objetivo es muy simple, mostrar en la pantalla el dato que se recibe serialmente y que se supone corresponde a la tecla que se ha presionado.

Para recibir los datos en la computadora se escribió un pequeño programa en Turbo C, cuya visualización se muestra en la **figura 19.5** y su código fuente se muestra en la **figura 19.6.** Éste se encarga de configurar el puerto serial para una velocidad de 1.200 bps, datos de 8 *bits*, sin paridad y con un *bit* de parada (*stop bit*).

Después de esto se queda en un ciclo donde verifica la llegada de un dato, y si este aparece, lo muestra en la pantalla y después continua verificando la línea nuevamente.

El programa listado recibe los datos a través del puerto COM1; si desea trabajar con el COM2 debe utilizar un adaptador de 9 a 25 pines que se consigue fácilmente en el mercado y además hacer un pequeño cambio en el programa en la parte inicial donde se selecciona el puerto de comunicación.

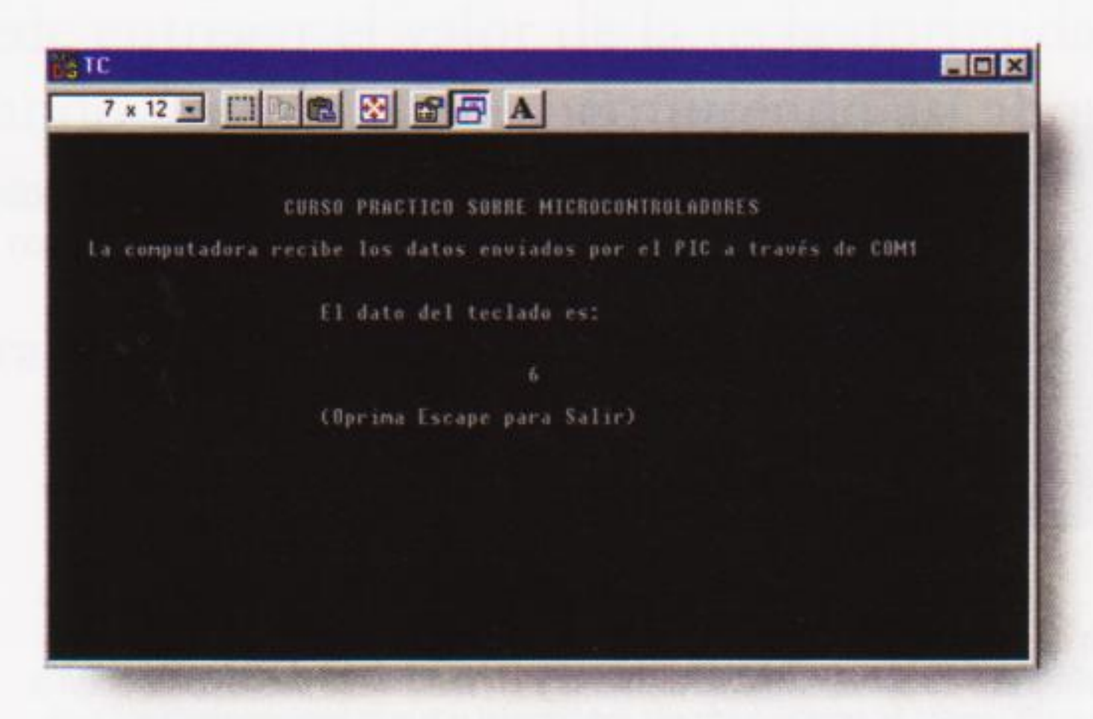


Figura 19.5 *Visualización del dato leído*

```
/* LA COMPUTADORA RECIBE LOS
DATOS SERIALES ENVIADOS POR
EL PIC  */
#include <conio.h>
#include <stdio.h>
#include <dos.h>
#include <math.h>
#include <bios.h>

  int puerto,COM1,COM2;
  int k,j,dato;
/*definición de variables*/
  int config;
  int COM1,COM2;
  char lectura[1];
  char dato1[2];

char leer()
{
    do{

  dato=bioscom(2,0x83,puerto);
/*leer dato recibido*/
      }
while(((dato<47)|(dato>72))&(!kbhit()));
    return(dato);
 }

void main(void)
{
  COM1=0;
  COM2=1;
  puerto=COM1;
/* definir cual puerto se
utiliza */

  clrscr();/*limpia pantalla*/
  config=0x83;  /*configurar
puerto: 1200 baudios,dato de
8 bits,no paridad, 1 bit de
parada*/

  bioscom(0,config,puerto);
/*configuración de los
puertos*/

gotoxy(20,4);
  printf(«CURSO PRÁCTICO
SOBRE MICROCONTROLADORES»);

  gotoxy(4,6);
  printf(«La PC recibe por COM1»);

  gotoxy(23,14);
  printf(«(Oprima Escape para Salir)»);
  gotoxy(23,9);
  printf(«El dato del teclado es:»);

do{
    if(!kbhit())
dato1[0]=leer();
    if(!kbhit())
    {
      gotoxy(40,12);
      printf(«%1s «,dato1);
    }

  }while(!kbhit());
  clrscr();
}
```

Figura 19.6 *Código fuente de la lectura del puerto serial compilado en Turbo C*

Ensamblaje del proyecto

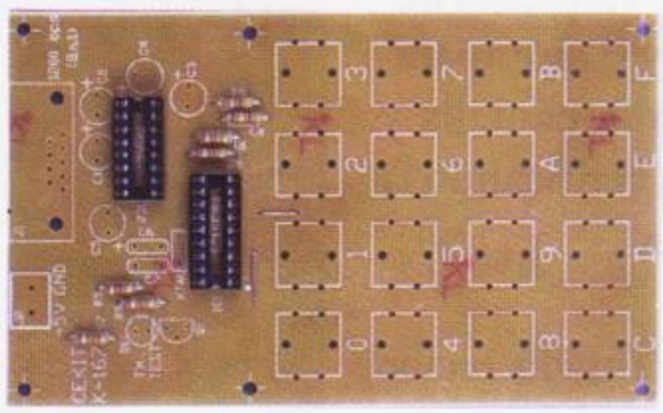
Figura 19.7. Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el teclado 4X4 con salida serial.

Con este sencillo proyecto usted podrá leer desde su computadora personal, y a través del puerto serial (COM1 ó COM2) la decodificación de cada una de las teclas del teclado hexadecimal con configuración matricial, incluido en la tarjeta del proyecto. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

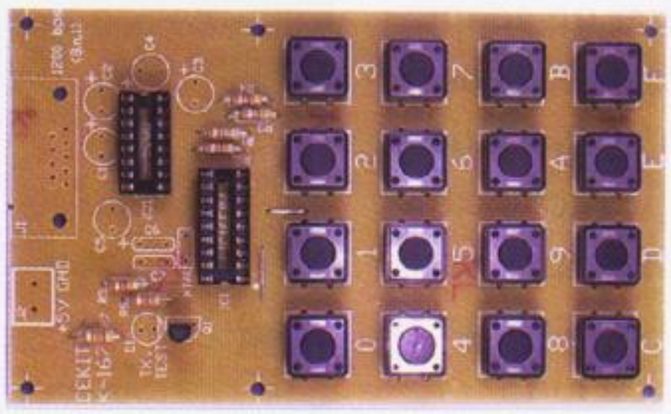
LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F84, grabado
1	IC2	Circuito integrado MAX232
1	Q1	Transistor NPN de propósito general 2N3904
1	D1	LED rojo de 5mm
5	C1, C2, C3, C4 y C5	Condensadores de electrolíticos de 10µF - 35V
2	C6 y C7	Condensadores cerámicos de 22 pF - 50V
6	R1, R2, R3, R4, R5 y R6	Resistencias de 2,7k - 1/4W
1	R7	Resistencia de 330 - 1/4W
1	XTAL	Cristal de cuarzo de 4MHz
1	K-167	Circuito impreso CEKIT
1	PROGRAMA K-167	Disquete con el programa del teclado
1	J1	Conector DB-9 hembra para circuito impreso
1	J2	Conector de tornillo de 2 pines
16		Pulsadores grandes para circuito impreso de 4 pines
1		Conector DB-9 hembra con carcasa
1		Conector DB-9 macho con carcasa
1		Base para circuito integrado de 18 pines
1		Base para integrado de 16 pines
1 m		Cable ribbon de 9 hilos

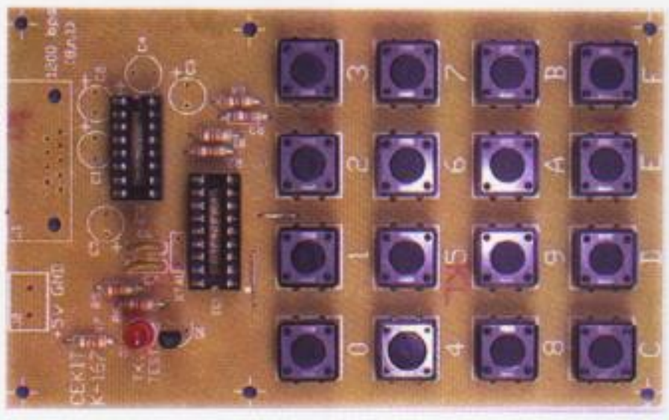
Paso 1
Instale y suelde los puentes de alambre, las resistencias R1 a R7 y las dos bases para los circuitos integrados IC1 e IC2.
Figura 19.8

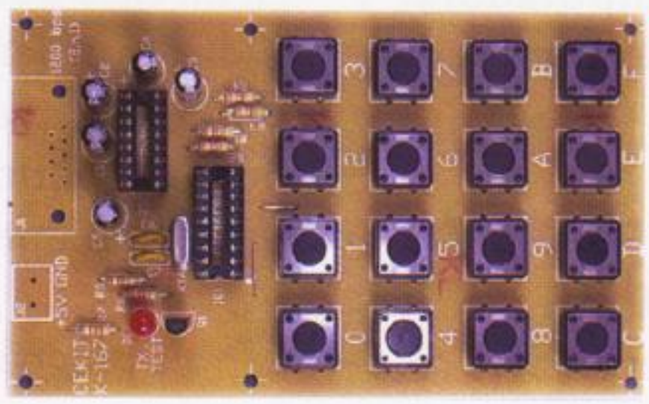


Paso 2
Luego instale y suelde el transistor NPN Q1 y las 16 teclas que conforman el teclado hexadecimal.
Figura 19.9

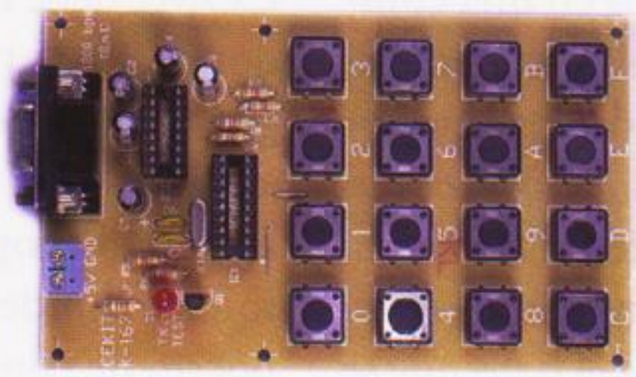


Paso 3
Instale y suelde el diodo LED indicador de transmisión y los condensadores cerámicos C6 y C7.
Figura 19.10

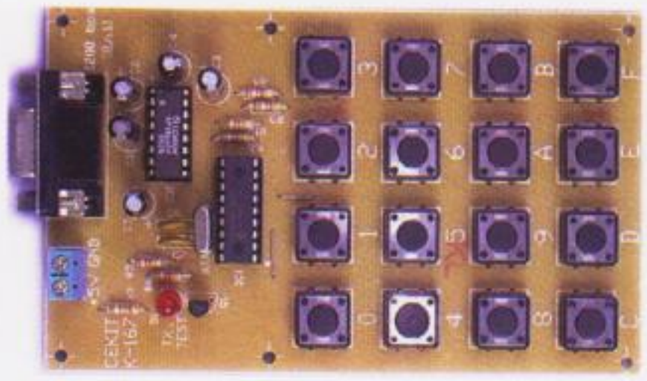




Paso 4
Instale y suelde el cristal de cuarzo de 4MHz XTAL y los cinco condensadores electrolíticos de 10 µF, C1 a C5.
Figura 19.11



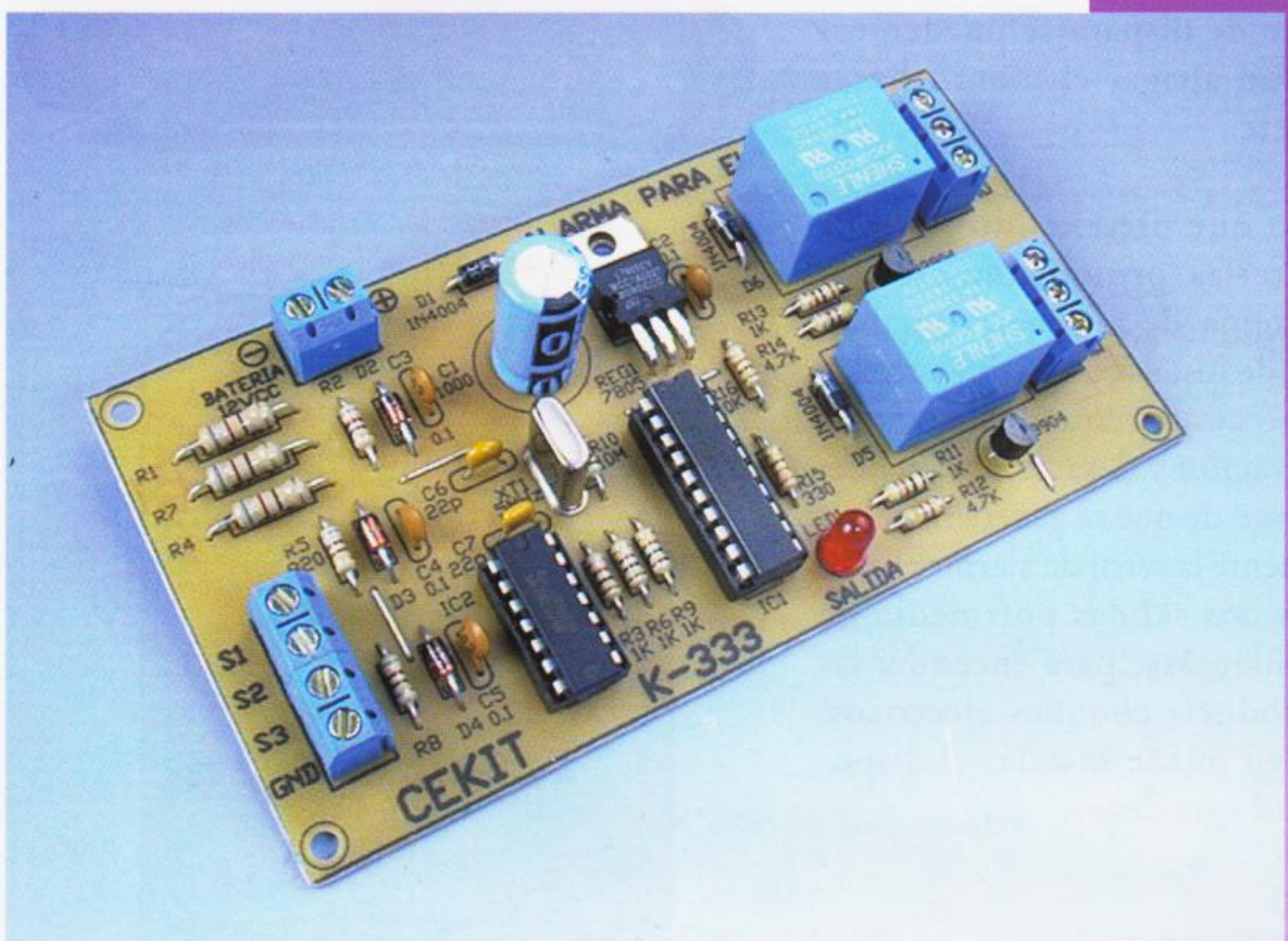
Paso 5
Instale y suelde el conector serial DB-9 para el puerto de la computadora y el conector de alimentación del circuito J2.
Figura 19.12



Paso 6
Aspecto final del teclado 4x4 con salida serial.
Figura 19.13

Paso 6. Pruebas finales
Para alimentar el circuito, en el conector tipo tornillo de dos pines, se debe proporcinar un voltaje de 5 voltios. Cada vez que se oprima una de las teclas el dato es transmitido, el diodo LED se activa momentáneamente y el resultado se puede apreciar en la pantalla de la computadora. Si presenta alguna anomalía, revise la construcción del cable serial y que el microcontrolador esté debidamente programado.

PROYECTO 20



ALARMA PARA EL AUTOMÓVIL

Este proyecto le permite construir un sistema de seguridad práctico y económico para ser instalado en un automotor. El microcontrolador le proporciona dos salidas de alarma para que las utilice como control sobre cualquier mecanismo; el límite es su imaginación.

Características

- Entrada para sensores de puertas abiertas
- Entrada para sensores de puertas aseguradas
- Entrada para sensores de automotor funcionando
- Salida por medio de dos relés
- Diodo LED indicador
- Microcontrolador Motorola

En estos tiempos de inseguridad cualquier precaución que se tome no está de más. Comercialmente se pueden conseguir muchos sistemas de alarma para automotores, que se activan mediante una señal de radio, cuentan con sensores infrarrojos, de vibración, etc., y al momento de dispararse pueden tomar acciones como apagar el motor, hacer sonar la bocina, etc.

El dispositivo que presentamos en esta ocasión se caracteriza principalmente por constituir un sistema de alarma confiable, económico y fácil de instalar en cualquier vehículo. A pesar de no contar con un sistema de radio para su activación y desactivación, puede utilizarse sin riesgo de que se genere involuntaria o accidentalmente la señal de alarma. En él están disponibles dos salidas por medio de relé que pueden adaptarse para encender las luces, sirenas, producir choques eléctricos, etc., es decir, usted puede instalar el dispositivo de seguridad que desee.

Funcionamiento

La tarjeta cuenta con un conector en el cual se instalan los sensores que deben informar sobre el estado de los distintos puntos del automotor y originar las señales que, de acuerdo a ciertas condiciones, deben producir la activación y desactivación de la alarma. En la **figura 20.1** puede apreciar la ubicación de los sensores en el automotor al igual que algunos posibles lugares para instalar la tarjeta.

Cuando se desconecta el motor y se alimenta el circuito, se inicia la actividad de la alarma y el diodo LED se enciende para indicar este estado del sistema.

A partir de ese momento se deben cumplir dos requisitos más para activar la alarma, ellos son: que las puertas se cierren y se les coloque el seguro. Cuando estas dos condiciones están dadas, se configura el armado del sistema, es decir la puesta en funcionamiento de la alarma.

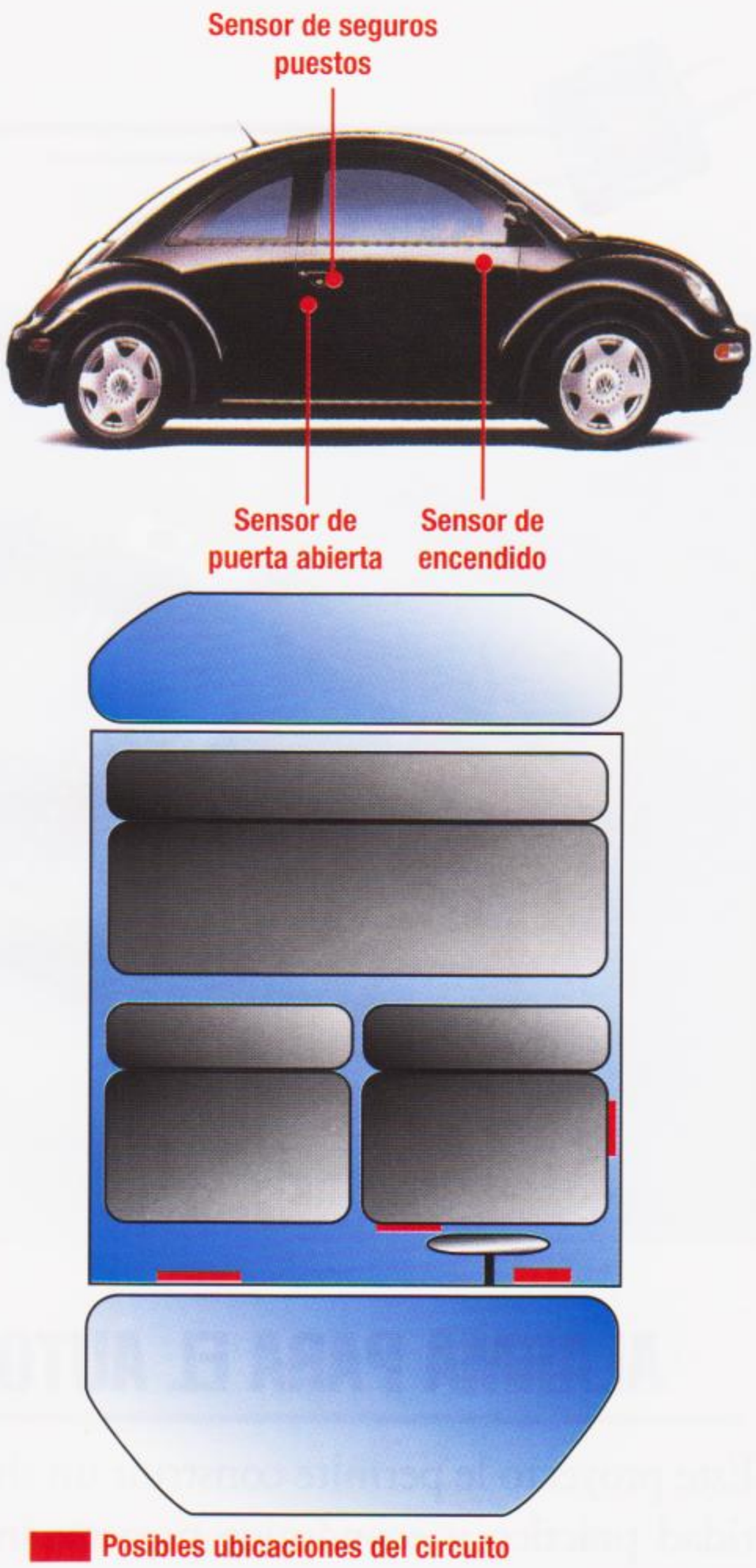


Figura 20.1 *Ubicación de los sensores dentro del vehículo*

Esto quiere decir que cuando apaga el motor, cierra las puertas y les coloca el seguro, la alarma se pone en estado de vigilancia de las puertas y puede abandonar tranquilamente el vehículo. Mientras permanece en este estado de vigilancia, el circuito se mantiene pendiente de los sensores de las puertas y hace parpadear el LED.

Recuerde que el circuito no cuenta con un sistema de radio para desactivar la alarma. ¿Cómo puede desactivarla entonces?

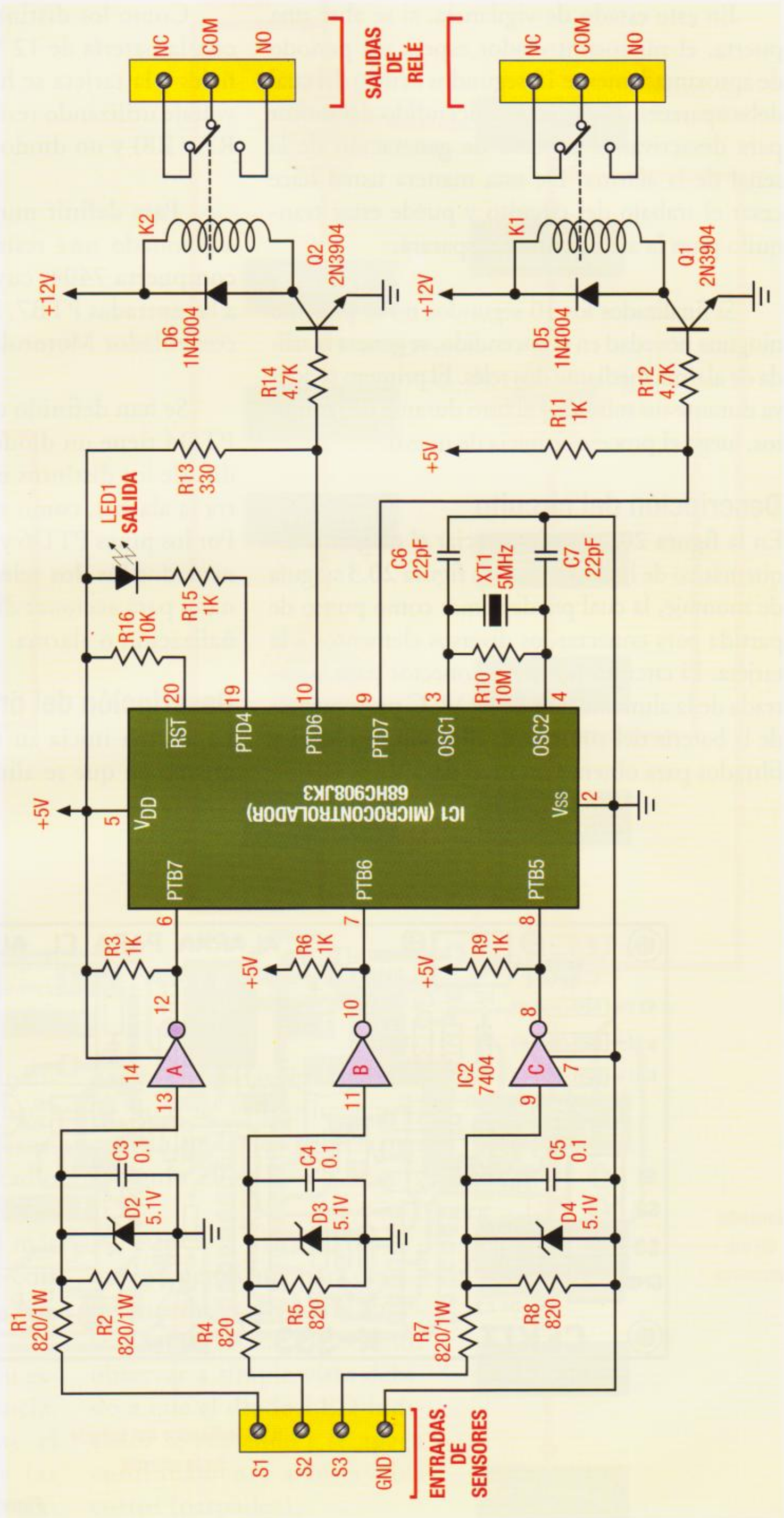


Figura 20.2 *Diagrama esquemático del circuito*

En este estado de vigilancia, si se abre una puerta, el microcontrolador espera un periodo de aproximadamente 10 segundos dentro del cual debe aparecer una señal de encendido del motor para desactivar el proceso de generación de la señal de la alarma. De esta manera usted hace cesar el trabajo del circuito y puede estar tranquilo pues la alarma no se disparará.

Si finalizados los 10 segundos no se presentó ninguna novedad en el encendido, se genera la salida de alarma mediante dos relés. El primero se activa durante un minuto y el otro durante dos minutos, luego el proceso se inicia de nuevo.

Descripción del circuito

En la **figura 20.2** puede apreciar el diagrama esquemático de la alarma y en la **figura 20.3** su guía de montaje, la cual puede tomar como punto de partida para conectar los diversos elementos a la tarjeta. El circuito posee un conector para la entrada de la alimentación de 12 VCC provenientes de la batería del automóvil; ellos son regulados y filtrados para obtener un nivel de 5 V.

Como los distintos sensores se alimentan con la batería de 12 V, para introducir sus señales a la tarjeta se han instalado divisores de voltaje utilizando resistencias (R1, R2, R4, R5, R7 y R8) y un diodo Zener de 5,1 V.

Para definir muy bien los voltajes, se ha adicionado una resistencia de *pull-up* y una compuerta 7404, cuya salida llega finalmente a las entradas PTB7, PTB6 y PTB5 del microcontrolador Motorola 68HC908jk3.

Se han definido tres pines de salida: por el PTD4 tiene un diodo LED que sirve para indicarle los distintos estados en que se encuentra la alarma, como se explicará más adelante. Por los pines PTD6 y PTD7 se encuentran conectados los dos relés que serán las salidas finales para accionar diversos dispositivos de señalización o alarma.

Descripción del programa

La alarma inicia su trabajo desde el instante mismo en que se alimenta el circuito, para lo

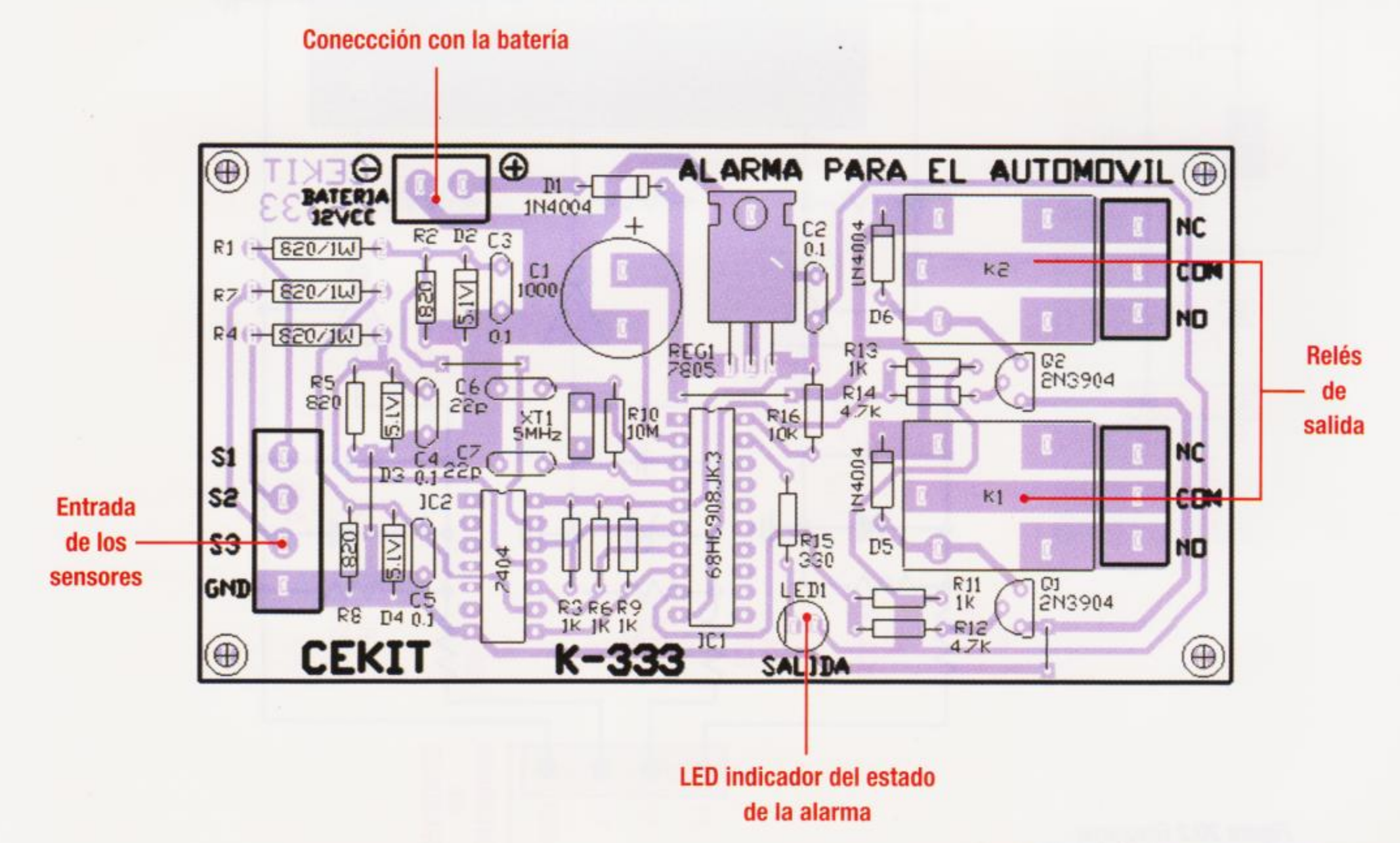


Figura 20.3 Guía de montaje de la tarjeta

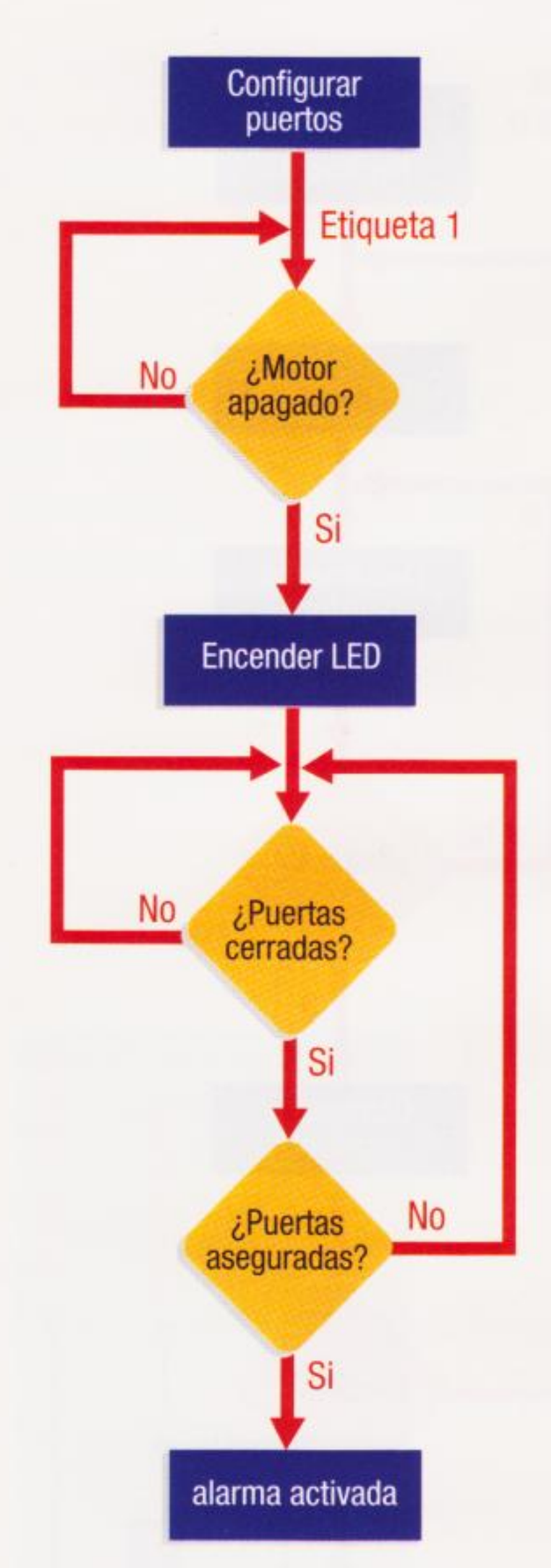


Figura 20.4 *Procedimiento de activación o habilitación de la alarma para que inicie su trabajo*

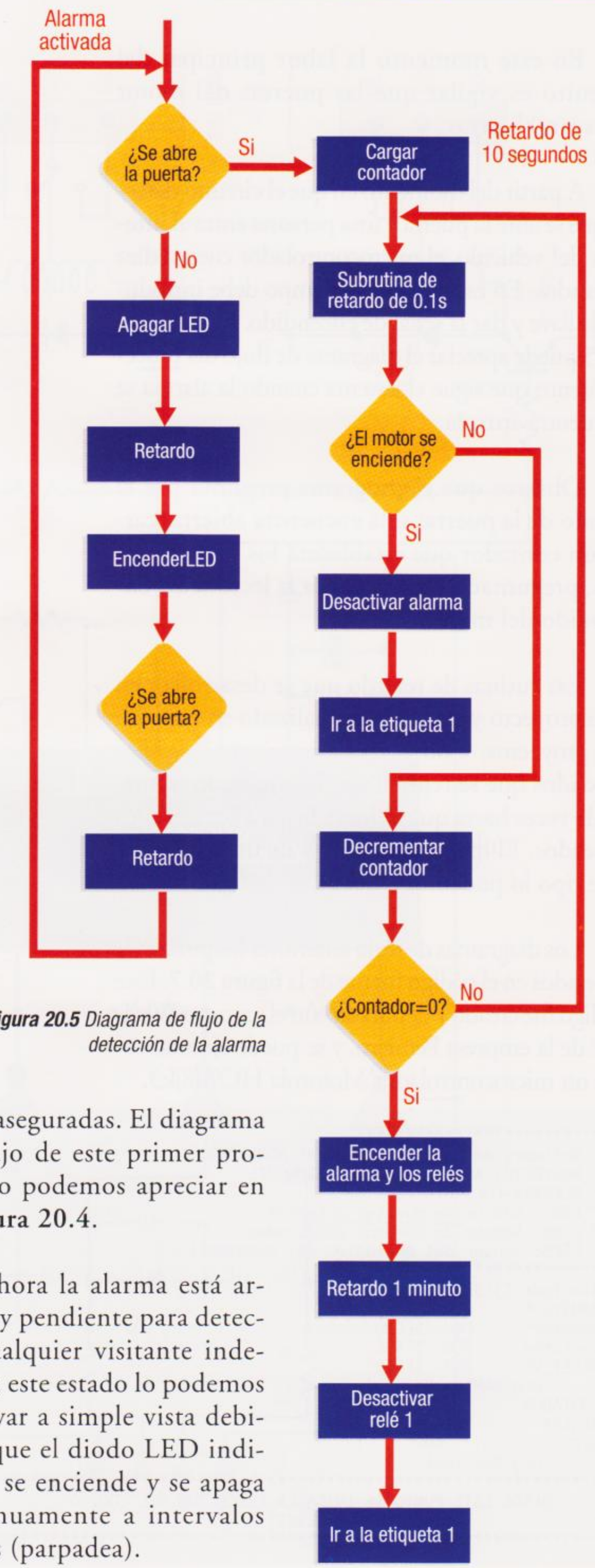


Figura 20.5 *Diagrama de flujo de la detección de la alarma*

cual se puede adicionar un interruptor en serie con la batería, este sencillo sistema reemplaza a los sistemas de radio de las alarmas comerciales y es una de las razones que más pesa para minimizar los costos de este sistema.

La alarma entra en su estado de armado o vigilancia, cuando de detecta que el motor está apagado y las puertas se encuentran cerradas y aseguradas. El diagrama de flujo de este primer proceso lo podemos apreciar en la **figura 20.4**.

Ahora la alarma está armada y pendiente para detectar cualquier visitante indeseado; este estado lo podemos observar a simple vista debido a que el diodo LED indicador se enciende y se apaga continuamente a intervalos cortos (parpadea).

En este momento la labor principal del circuito es vigilar que las puertas del motor no sean abiertas.

A partir del momento en que el circuito detecta que se abre la puerta y una persona entra al interior del vehículo, el microcontrolador cuenta diez segundos. En ese espacio de tiempo debe introducir la llave y dar la señal de encendido. En la **figura 20.5** puede apreciar el diagrama de flujo del procedimiento que sigue el sistema cuando la alarma se encuentra armada.

Observe que el programa pregunta por el estado de la puerta; si la encuentra abierta, carga un contador que establecerá los diez segundos, preguntado cada 0,1s por la lectura del encendido del motor.

Las rutinas de retardo que se desarrollan en este proyecto ya las hemos analizado en anteriores proyectos. Consisten básicamente en ciclos anidados que se repiten un determinado número de veces hasta que se logra llegar a los tiempos deseados. El diagrama de flujo de un retardo de este tipo lo podemos apreciar en la **figura 20.6**.

Los diagramas de flujo anteriores los puede ver reflejados en el código fuente de la **figura 20.7**. Este código fue creado y depurado con el entorno WinIDE de la empresa Pemicro, y se puede ejecutar sobre un microcontrolador Motorola HC908jk3.

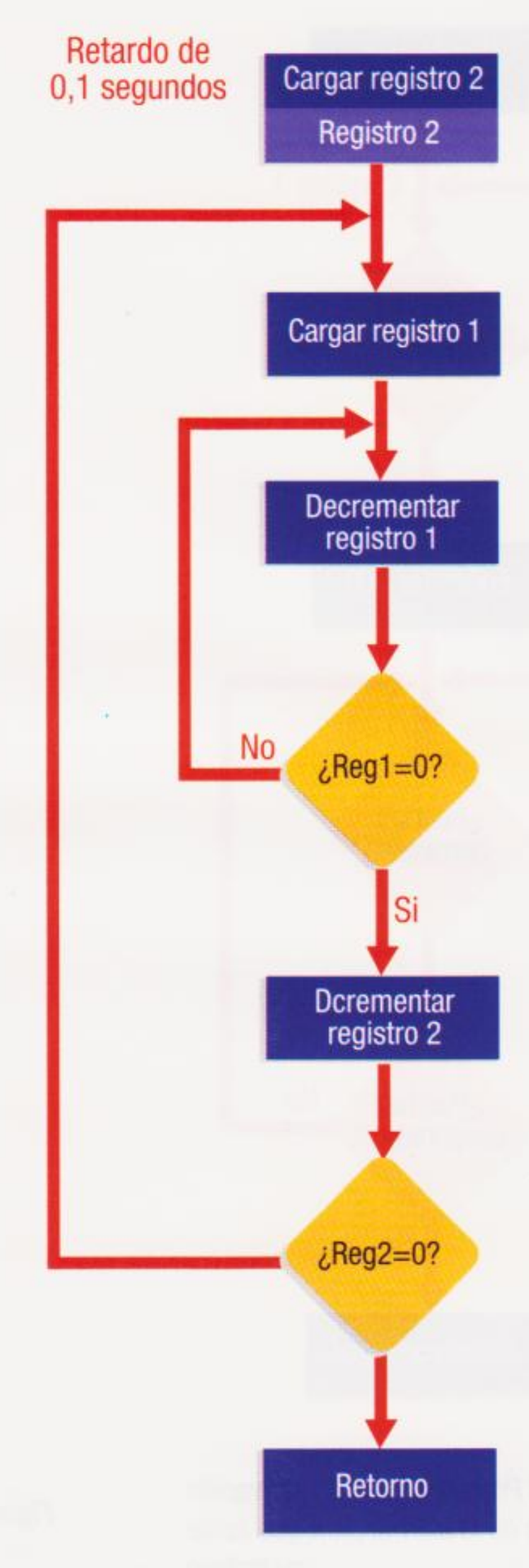


Figura 20.6 *Diagrama de flujo de la rutina de retardo*

```
************************
* Software en assembler para el HC08 JK3,
* PROYECTO: ALARMA PARA EL AUTOMOVIL
* PEREIRA-COLOMBIA-2001
* PTB7: Sencor de puertas abiertas
* PTB6: Sensor de puertas aseguradas
* PTB5: Sensor del encendido del automóvil
************************
$include 'jl3regsg.inc'
RAMStart      EQU  $0080
RomStart      EQU  $EC00   ;
Vectores      EQU  $FFDE
RESET_VEC     EQU  $FFFE
      org RAMStart
CONTADOR          RMB     1
REGIS1            RMB     1
REGIS2            RMB     1
      org RomStart
*************************************************************
*  - DESDE ESTE PUNTO SE INICA LA EJECUCIÓN DEL CÓDIGO*
*              DESPUÉS DE UN RESET.                        *
*************************************************************
```

```
INICIO:
        RSP
        Bset COPD,CONFIG1                ; deshabilita el COP (watchdog)
        CLRA
        CLRX
        MOV #$ff,DDRD                    ; CONFIGURO EL PUERTO D COMO SALIDAS
        MOV #$000,DDRB                   ; CONFIGURO EL PUERTO A COMO ENTRADAS
INI:
        CLR      PORTD
        BSET     4,PORTD
        RSP
ESPERA1:
        BRCLR    5,PORTB,ESPERA1         ; ESPERA QUE SE APAGUE EL MOTOR
                                         ; PARA INICIAR EL PROGRAMA
                                         ; AL ACTIVARSE, SE ENCIENDE EL LED
        BCLR     4,PORTD
                                         ; AHORA DEBE ESPERAR QUE LAS PUERTAS SE ASEGUREN LAS PUERTAS

ESPERA2:
        BRSET    6,PORTB,ESPERA2
        BSET     4,PORTD
                                         ; AHORA HAY QUE ESPERAR QUE LA PUERTA SE ABRA
                                         ; Y HAGA PARPADEAR EL LED
ESPERA3:
        RSP
        BRCLR    7,PORTB,RET10S
        BCLR     4,PORTD                 ; APAGA EL LED
        JSR      RETA01                  ; RETARDO DE 0,5 SEGUNDOS
        JSR      RETA01
        JSR      RETA01
        JSR      RETA01
        BSET     4,PORTD                 ; ENCIENDO EL LED
        BRCLR    7,PORTB,RET10S
        JSR      RETA01
        JSR      RETA01
        JSR      RETA01
        BRA      ESPERA3
RET10S:
        MOV      #$64,CONTADOR           ; 100 VECES .1 SEG. = 10 SEG
LAZO10  JSR      RETA01
        BRCLR    5,PORTB,INI ;
        DBNZ     CONTADOR,LAZO10
        BSET     6,PORTD                 ; ACTIVAR LA ALARMA
        BSET     7,PORTD
        BCLR     4,PORTD
        JSR      RET20S                  ; RETARDO DE 1 MINUTO
        JSR      RET20S
        JSR      RET20S
        BCLR     6,PORTD
        JSR      RET20S                  ; RETARDO DE 1 MINUTO
        JSR      RET20S
        JSR      RET20S
        BCLR     7,PORTD
        BSET     4,PORTD
        BRA      INI
*****************************************************************
* RETARDO- SUBRUTINA DE RETARDO POR SOFTWARE *
*****************************************************************
RETA01
        LDA     #128T ; RETARDO DE 0.1 s
ET2     CLRX
ET3     DECX
        BNE      ET3
        DECA
        BNE      ET2
        RTS
RET20S
        LDA     #20T
        STA     REGIS1
ET1     JSR     RETA01
        DEC     REGIS1
        BNE     ET1
        RTS
        Org RESET_VEC
        Dw INICIO                        ;al darse reset salta a Start
        END
```

Figura 20.7 *Código fuente del proyecto*

Ensamblaje del proyecto

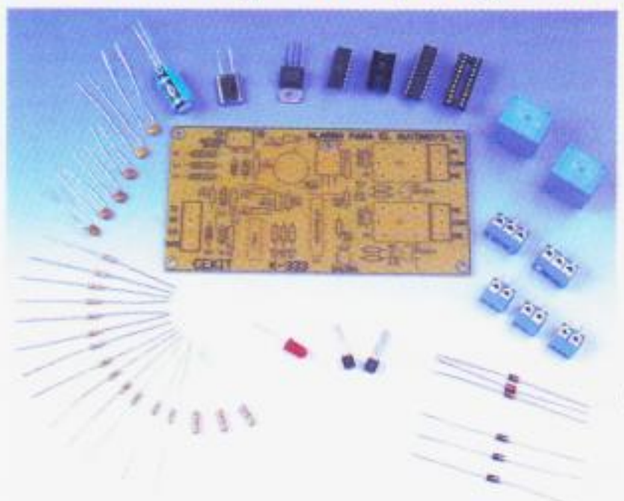


Figura 20.8 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar la alarma para el automóvil.*

Instalando este sencillo circuito dentro de su vehículo, usted podrá protegerlo contra robo, y vigilar su acceso a través de las puertas por medio de tres sensores. También podrá bloquear el encendido del motor con un contacto normalmente abierto o normalmente cerrado, con capacidad de 10A disponible en la salida de la alarma. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

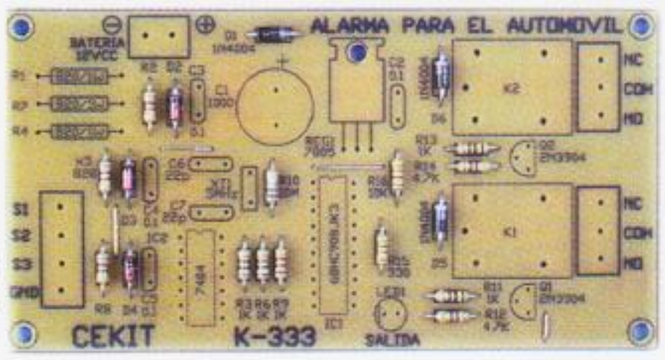
LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador 68HC908JK3, programado
1	IC2	Circuito integrado inversor 7404
1	REG1	Regulador fijo de +5V -1A, 7805
2	Q1 y Q2	Transistores de propósito general NPN, 2N3904
3	D2 a D4	Diodos Zener de 5.1V a 1W
3	D1, D5 y D6	Diodos rectificadores de propósito general 1N4004
1	LED1	Diodo LED rojo de 5mm
1	C1	Condensador electrolítico de 1000uF - 25V
4	C2 a C5	Condensadores cerámicos de 0.1uF - 50V
2	C6 y C7	Condensadores cerámicos de 22pF - 50V
1	XT1	Cristal de quarzo de 5MHz
3	R1, R4 y R7	Resistencias de 820 - 1W
3	R2, R5 y R8	Resistencias de 820 - 1/4W
5	R3, R6, R9, R11 y R13	Resistencias de 1k - 1/4W
1	R10	Resistencia de 10M - 1/4W
2	R12 y R14	Resistencias de 4.7k - 1/4W
1	R15	Resistencia de 330 - 1/4W
1	R16	Resistencia de 10k - 1/4W
2	K1 y K2	Relés de 12VCC
1		K-333 Circuito impreso CEKIT
2		Conectores de tornillo de 3 pines
3		Conectores de tornillo de 2 pines
1		Base para circuito integrado de 20 pines
1		Base para circuito integrado de 14 pines

Paso 1

Instale y suelde los puentes de alambre, las trece resistencias de 1/4W, los diodos zener D2, D3 y D4 y los diodos rectificadores D1, D5 y D6.

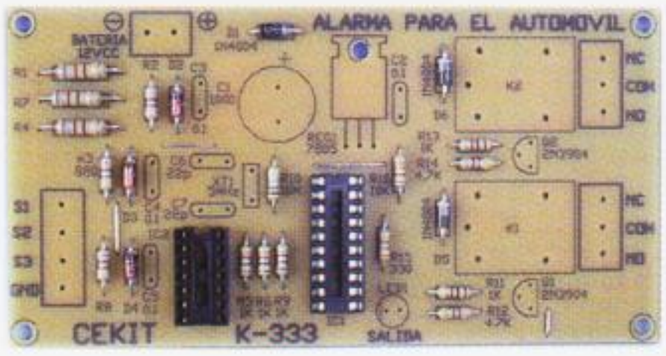
Figura 20.9



Paso 2

Luego instale y suelde las bases para los circuitos integrados IC1 e IC2, y las resistencias de 1W R1, R4 y R7.

Figura 20.10



Paso 3

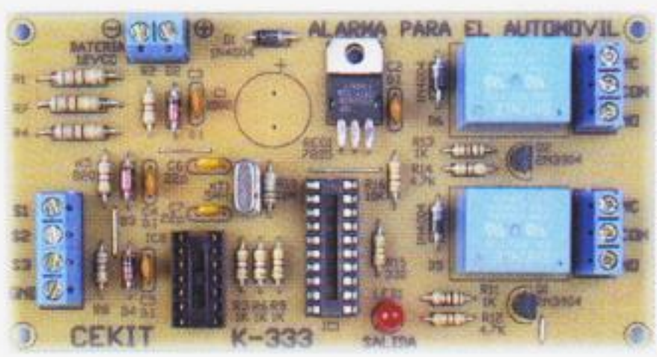
Instale y suelde el diodo LED indicador de salida y los transistores NPN de propósito general Q1 y Q2.

Figura 20.11

Paso 4

Instale y suelde el regulador de voltaje integrado de +5V REG1, y todos los conectores de entrada y salida de la alarma.

Figura 20.12



Paso 5

Instale y suelde los condensadores cerámicos C2 a C7, el cristal de cuarzo de 5MHz XT1, y los dos relés de salida K1 y K2.

Figura 20.13

Paso 6

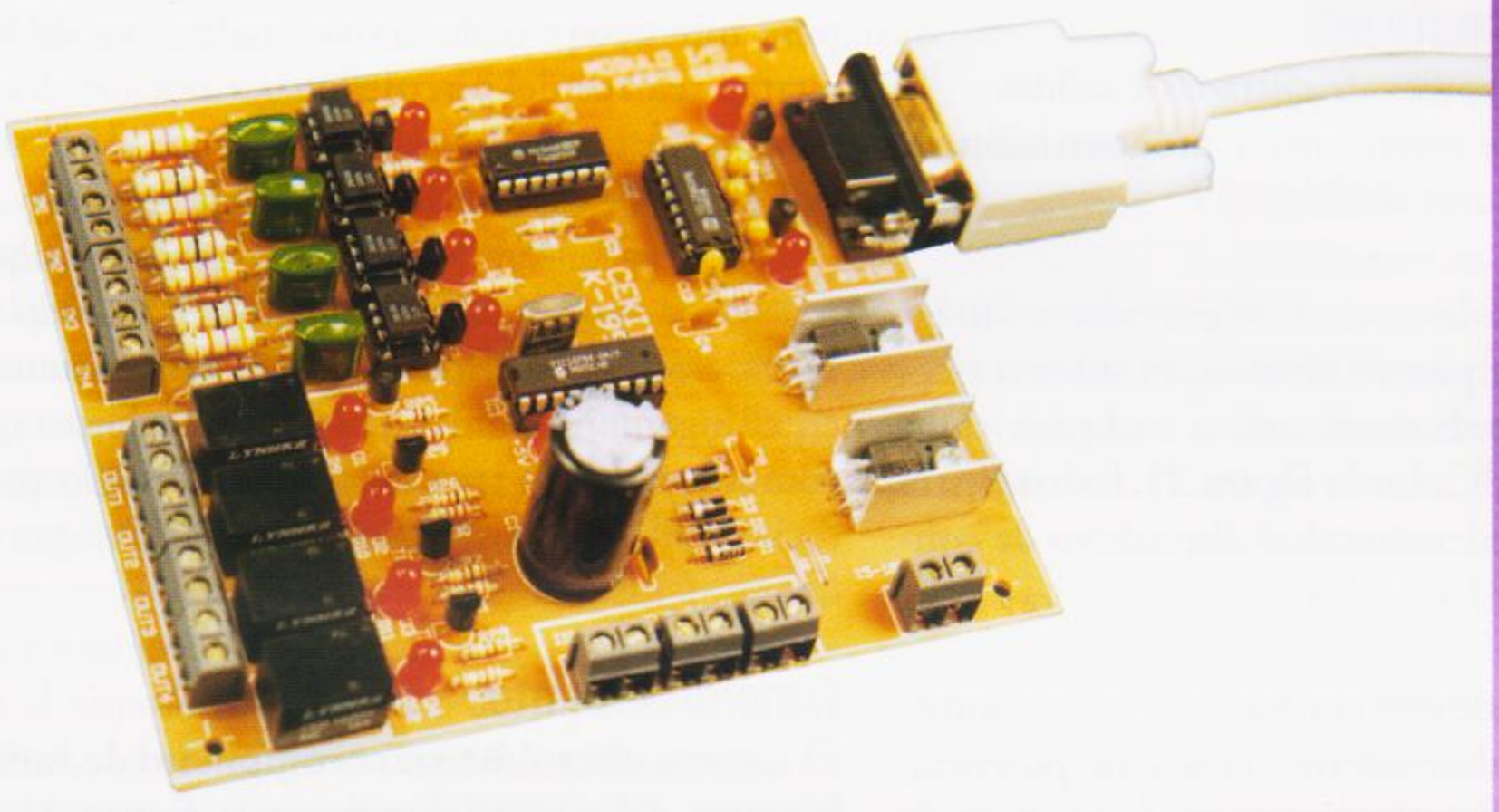
Aspecto final del circuito de la alarma luego de soldar el condensador electrolítico C1 y colocar los circuitos integrados en sus bases.

Figura 20.14

Paso 6. Pruebas finales

Para alimentar el circuito utilice el conector tipo tornillo de dos pines que se ve en la parte superior de la tarjeta. Conecte en este par de puntos los terminales positivo y negativo de la batería, teniendo cuidado en conectar con la polaridad correcta como se indica en el screen de la tarjeta. Los sensores pueden funcionar con el mismo voltaje de la batería y se asume que, cuando un sensor se cierra, al circuito le entran 12 V, y cuando el sensor está abierto a la entrada de la tarjeta se tienen 0 V.

PROYECTO 21



CONTROL I/O DE POTENCIA POR EL PUERTO SERIAL

Construya este interesante proyecto para ejercer el monitoreo de entradas y el control de salidas de potencia, a través del puerto serial de la computadora.

Características del sistema electrónico

- Cuatro salidas por relevo de 5 amperios
- Cuatro entradas de 24-60 VCC ó 50-220 VCA, aisladas por optoacoplador
- Diodos LED indicadores del estado de las entradas y las salidas
- Caja metálica para su instalación en la pared
- Permite la entrada y salida de cables por tuberías
- Control del sistema por comunicaciones RS-232
- Alimentación de 110 VCA ó 220 VCA

Características del programa

- Programación de secuencias automáticas
- Control y monitoreo por el puerto serial
- Interfaz bajo Windows
- Control manual por medio de clics en el ratón
- Lenguaje de programación parecido al de los PLC
- No esclaviza la PC mientras ejerce el control de I/O

El sistema consta de:

- Tarjeta electrónica de entradas y salidas
- Programa de monitoreo y programación de secuencias automáticas

El módulo de control de entradas y salidas de potencia por puerto serial es un sistema electrónico que puede conectarse a cualquier computadora tipo PC. En la **figura 21.1** se muestra la forma como se conecta el dispositivo de control a una PC.

Por medio de este sistema es posible controlar electroválvulas, relevos de mayor potencia, contactores para activar motores, luces, paso de fluidos, hornos, y, en fin, cualquier dispositivo eléctrico o electrónico que pueda ser manejado a través de contactos de relevo de 5 amperios.

Adicionalmente, el control de dichos relevos puede condicionarse a determinados eventos que pueden ser capturados por sensores que suministren señales ON/OFF, tales como reflectores ópticos, interruptores mecánicos, magnéticos o capacitivos, sensores ultrasónicos, etc. La única condición que deben tener los sensores, para que el módulo I/O pueda capturarlos en sus entradas, es que entreguen un voltaje de CC desde 20 hasta 50 voltios, o un voltaje de CA desde 60 hasta 220 voltios.

Mediante el programa de monitoreo, escrito en el lenguaje de programación Visual Basic, es posible ver cuáles entradas se encuentran activas, y a la vez, conocer el estado de las salidas que la computadora haya ordenado activar. Con este mismo programa, con sólo hacer clic sobre una de las cuatro salidas, usted ordenará a la tarjeta electrónica, que puede estar instalada en un sitio diferente al de la PC, para que active o desactive cualquiera de los cuatro relevos del circuito.

La distancia entre la tarjeta electrónica y la PC puede alcanzar hasta los 200 metros, aunque este valor cambia notablemente de acuerdo al ruido eléctrico y al ruido magnético existentes en el lugar, a la calidad del cable usado para la comunicación, al tipo de conducto usado para dicho cable, y a muchos más parámetros que es difícil mencionarlos.

Funcionamiento

El sistema electrónico está compuesto de varios bloques, tal como lo muestra la **figura 21.2**. Aunque todos los bloques son indispensables, el corazón del sistema es el microcontrolador PIC, ya que es este circuito integrado quien se comunica con la computadora, quien detecta el estado de las entradas y quien ordena el encendido de los relevos de las salidas. Veamos a continuación cómo funciona cada una de las secciones del proyecto.

Fuente de poder. Este bloque es el que se encarga de suministrar los niveles de voltajes apropiados, tanto para los circuitos digitales, como para los relevos que usan voltajes diferentes. En la **figura 21.3** puede ver el diagrama esquemático de la fuente de voltaje usa-

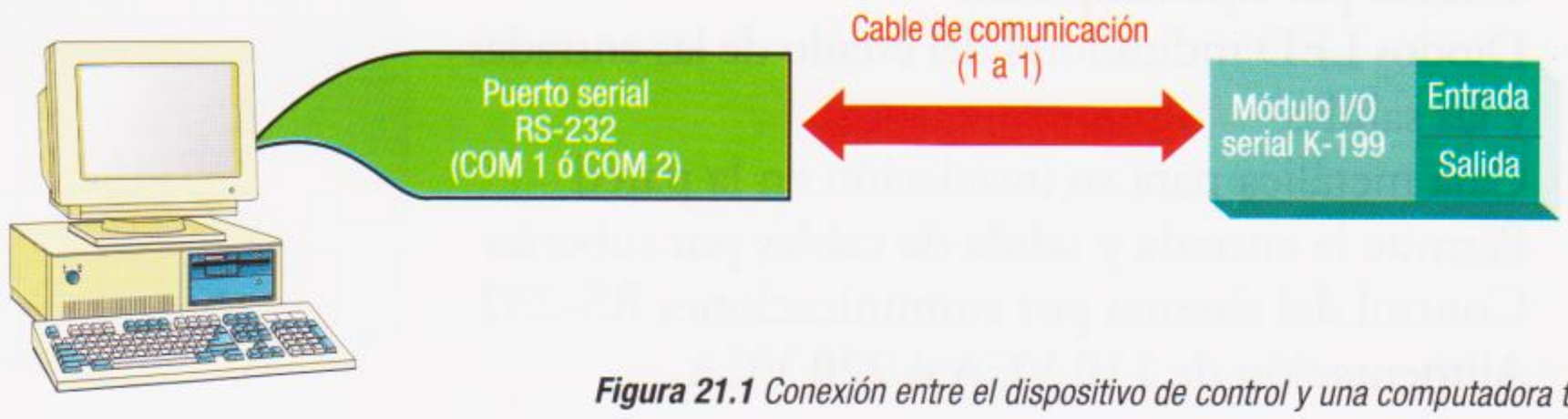


Figura 21.1 *Conexión entre el dispositivo de control y una computadora tipo PC*

Figura 21.2 Diagrama de bloques del sistema electrónico de control

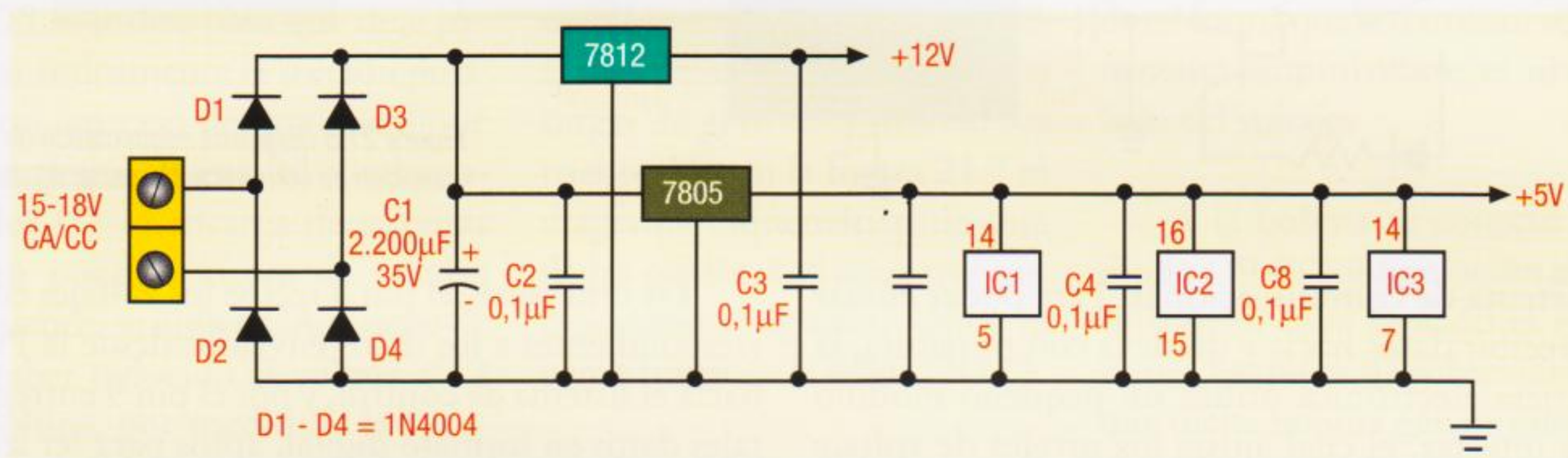


Figura 21.3 Diagrama esquemático de la fuente de poder

da para el circuito. Solamente se hace uso de 5 voltios para alimentar todos los circuitos, a excepción de los relevos que usan 12 voltios para activar sus bobinas.

Sistema de control. El sistema de control de entradas y salidas, y de las comunicaciones está a cargo del IC1, un microcontrolador PIC 16F84. Observe en la **figura 21.4** el diagrama esquemático de esta sección del proyecto. Como puede ver, los cuatro primeros *bits* del puerto A han sido utilizados para las salidas, mientras que las entradas van hacia el puerto B a sus 4 últimos *bits*. De este mismo puerto, los *bits* 0 y 1 se han diseñado de tal manera que sirvan para las comunicaciones con la computadora, uno como receptor y el otro para la transmisión de datos, respectivamente.

Las funciones del microcontrolador son, específicamente: capturar los datos de las cuatro entradas, enviar las señales de encendido hacia las cuatro salidas por relevo, y la recepción y transmisión de los datos desde y hacia la PC, correspondientes a los estados de las cuatro salidas y de las cuatro entradas.

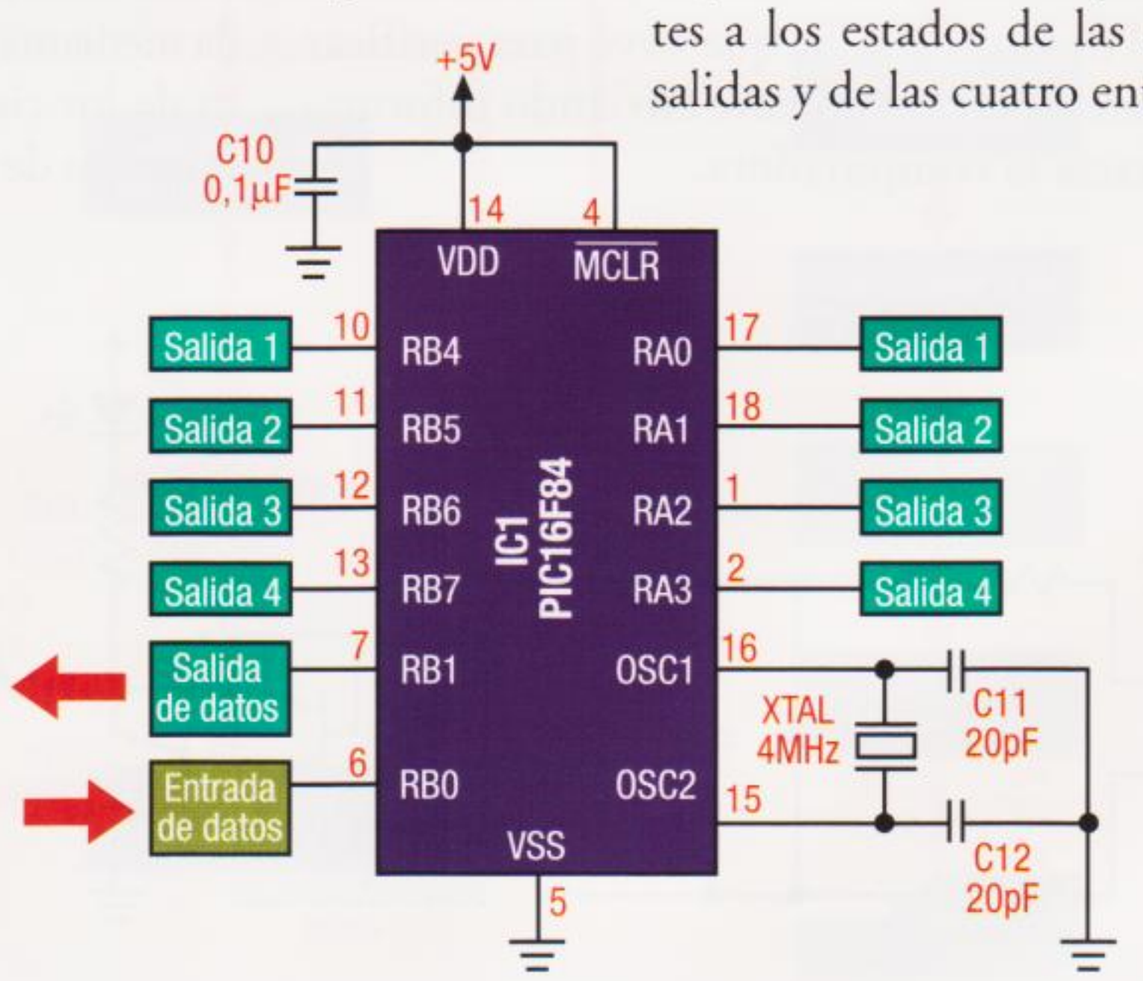


Figura 21.4 Diagrama esquemático del sistema de control con microcontrolador PIC16f84

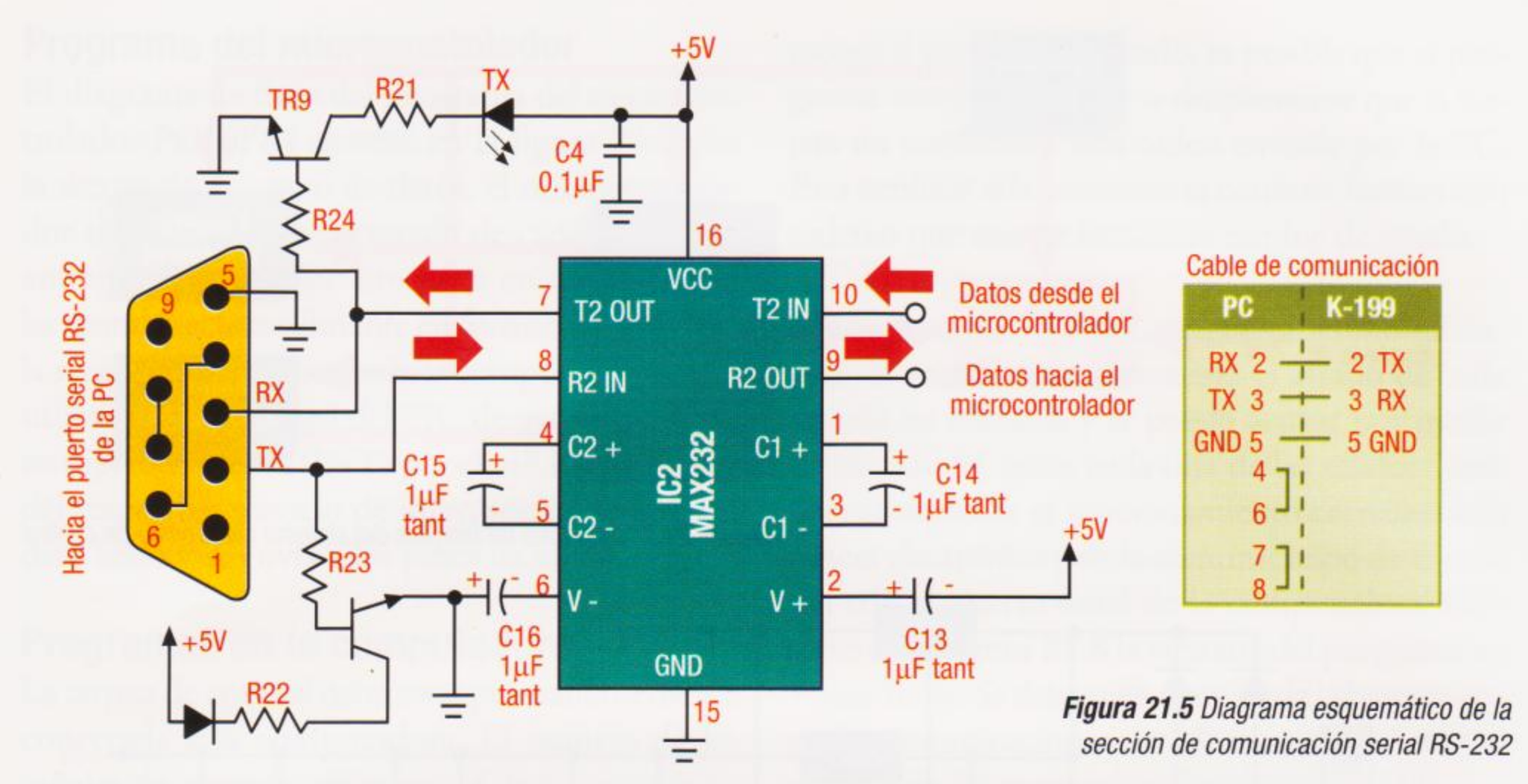


Figura 21.5 Diagrama esquemático de la sección de comunicación serial RS-232

Sistema de comunicaciones. Para poder enviar y recibir datos hacia y desde la computadora, la tarjeta electrónica utiliza un pequeño módulo de interfaz, el cual ajusta los niveles de voltaje digital a los necesarios en una comunicación serial RS-232. Observe en la **figura 21.5** el diagrama esquemático del sistema de interfaz RS-232.

El circuito integrado IC3 toma por el pin 10 los datos digitales provenientes del microcontrolador y los envía por el pin 7 hacia el conector serial. A este mismo pin se ha conectado el diodo LED Tx, que sirve para verificar si el sistema está realmente enviando información hacia la computadora.

De otra parte, el pin 8 recibe los voltajes correspondientes a los datos enviados desde la PC hacia el sistema de control, y por el pin 9 entrega tales datos en formato digital, aptos para ser leídos por el microcontrolador. A la entrada de datos provenientes de la computadora, también se ha instalado el diodo LED Rx para indicar el estado de la línea de recepción de datos.

Entradas. El sistema consta de cuatro entradas que se activan por medio de los voltajes CA o CC descritos anteriormente. Cada una de ellas está aislada mediante optoacopladores que protegen el resto de los circuitos y a la computadora de sobre impulsos de energía o de equivocaciones en el ni-

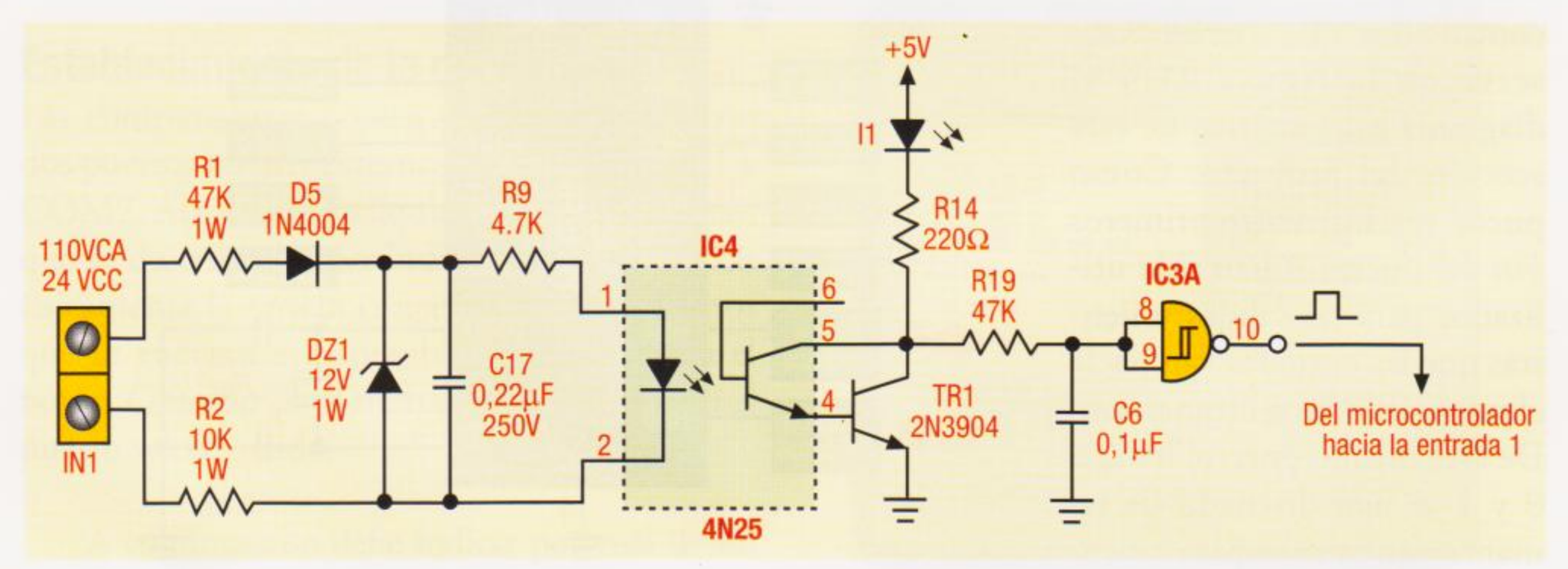


Figura 21.6 Diagrama esquemático de una de las entradas

vel de voltaje aplicado a dichas entradas. Vea en la **figura 21.6** el diagrama esquemático de una de las entradas. Las otras tres son exactamente iguales. Veamos como funciona la entrada IN1.

Cualquier voltaje que se aplique en los bornes de entrada se reducirá a 12 voltios, gracias a las resistencias R1 y R2 y al diodo Zener DZ1. El diodo D5 se utiliza para que deje pasar únicamente la sección positiva, en caso de que se aplique corriente alterna. El condensador C7 se encarga de suavizar un poco la curva de voltaje cuando se aplica CA. Luego de haber reducido el voltaje a 12 voltios, por medio de la resistencia R9 hacemos circular una pequeña corriente a través del optoacoplador. Hasta este punto existen conexiones físicas entre el circuito y las entradas.

Al otro lado del optoacoplador, la pequeña corriente entregada por su transistor es amplificada por el transistor externo TR1, que a su vez hace encender el LED indicador del estado de la entrada correspondiente. La señal es llevada luego a una compuerta NAND *schmitt trigger*, que, ayudada por la resistencia R19 y el condensador C6, corrige los niveles de la señal para evitar falsos disparos y para entregar voltajes digitales, que luego son llevados hacia el microcontrolador.

Salidas. Las cuatro salidas son a través de relevos con el fin de aislar eléctricamente el circuito de control y el dispositivo que se vaya a controlar. Los contactos de dichos relevos soportan una corriente de aproximadamente 5 amperios, suficiente para activar otros sistemas como los que mencionamos anteriormente.

Para cargas superiores deberán usarse relevos de mayor potencia, activados por los relevos de la tarjeta; o contactores, para el caso del control de motores o cargas de gran consumo de corriente. Vea en la **figura 21.7** el diagrama esquemático de una de las salidas. Las otras tres son exactamente iguales. Observe como funciona la salida OUT1.

La señal de control, enviada por el microcontrolador, es llevada a la base del transistor TR5 por medio de la resistencia R25. Este transistor entrega por medio de su colector el nivel de tierra, con el cual enciende el LED O1, que indica que dicha salida se encuentra activada. Este LED obtiene el nivel positivo a través de la resistencia R29. Con ese mismo nivel de tierra se energiza el relevo por uno de los terminales de su bobina. El otro extremo ya se encuentra conectado a 12 voltios directamente, de tal forma que el transistor sólo necesita suministrarle el nivel bajo del voltaje.

A la bobina se conectó el diodo D9 en configuración antiparalelo, cuya función es eliminar los picos de sobrevoltaje que dicha bobina genera cuando se desenergiza. Los bornes de la salida corresponden a los contactos NO (normalmente abiertos) del relevo.

Figura 21.7 *Diagrama de flujo del programa del microcontrolador*

Programa del microcontolador

El diagrama de flujo del programa del microcontrolador PC16F84 aparece en la **figura 21.7**. En la secuencia de envío de datos, el microcontrolador simplemente lee el estado de cada una de las entradas externas, las almacena en un registro y las manda secuencialmente en formato serial. Para la recepción de datos desde la computadora, se ha utilizado la capacidad del PIC de generar una interrupción externa (INT) por el pin RB0. Dentro del respectivo servicio de interrupción se arma el dato leído y se envía a los pines de salida.

Programas en la computadora

La tarjeta de control debe estar permanentemente conectada a la computadora. El manejo de las señales de entrada, así como el de las señales de salida, puede ser en forma manual o en forma automática. En el primer caso, solamente podemos estar observando el estado de las entradas y con el ratón podemos activar o desactivar cualesquiera de las cuatro salidas.

En el manejo automático, la computadora utiliza un programa que ejecuta otro programa determinado que haya escrito el usuario, de acuerdo a las necesidades del proceso que desea controlar. El desarrollo de la ejecución de una secuencia automática, a cargo de la computadora, tiene dos pasos: la escritura o edición del programa y la ejecución de dicho programa. Cualquiera de las opciones requiere de una previa conexión de la tarjeta electrónica de control, con el puerto serial de la PC.

Establecimiento de la comunicación

Las computadoras poseen generalmente uno o dos puertos seriales externos, llamados COM1 y COM2. Al ejecutar cualquiera de los programas, que se describen a continuación, debe conectar físicamente la tarjeta con uno de estos puertos que se encuentre disponible. Luego oprima el botón *Conectar* para enlazar el programa con el puerto seleccionado.

A continuación debe indicar por cuál de los puertos se ha hecho la conexión. Si en este punto escoge el puerto equivocado, es posible que el programa muestre un error o simplemente que la tarjeta no responda a una orden enviada por la PC. Para verificar si la conexión es correcta, bastará con ordenar que una de las salidas cambie de estado.

Modo manual. En este programa, como acabamos de mencionar, se muestra el estado de cada una de las entradas y se puede activar por medio de un clic del ratón cada una de las salidas. Sirve para comprobar el funcionamiento correcto de la tarjeta electrónica y de la comunicación de la misma con el puerto serial de la computadora. Observe en la **figura 21.8** la ventana del programa en Visual Basic. Se debe tener habilitada la herramienta de comunicaciones de Microsoft destinada al manejo de los puertos seriales (MSCOMM).

Modo de edición. Éste es el primer paso para el desarrollo de un programa que ejecuta una secuencia automática que dependa del estado de las entradas, de un determinado tiempo que haya transcurrido, y en fin, de una serie de circunstancias que hagan que

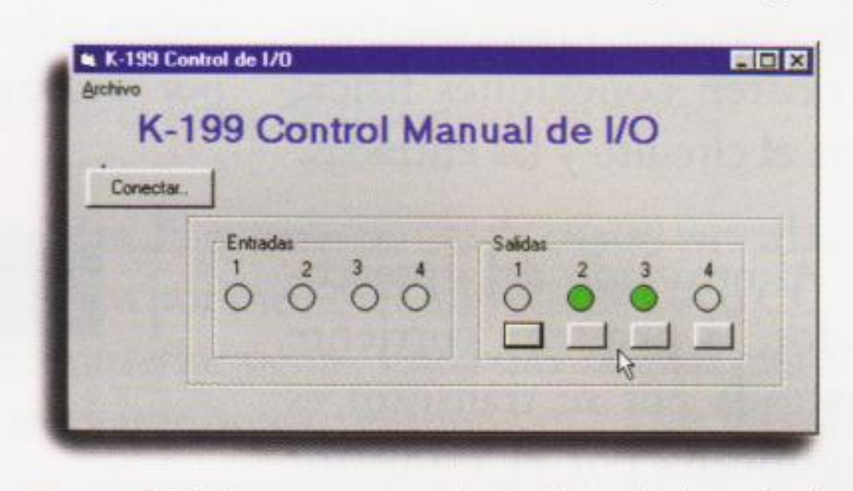


Figura 21.8 *Programa para el monitoreo de las entradas y el control manual de las salidas*

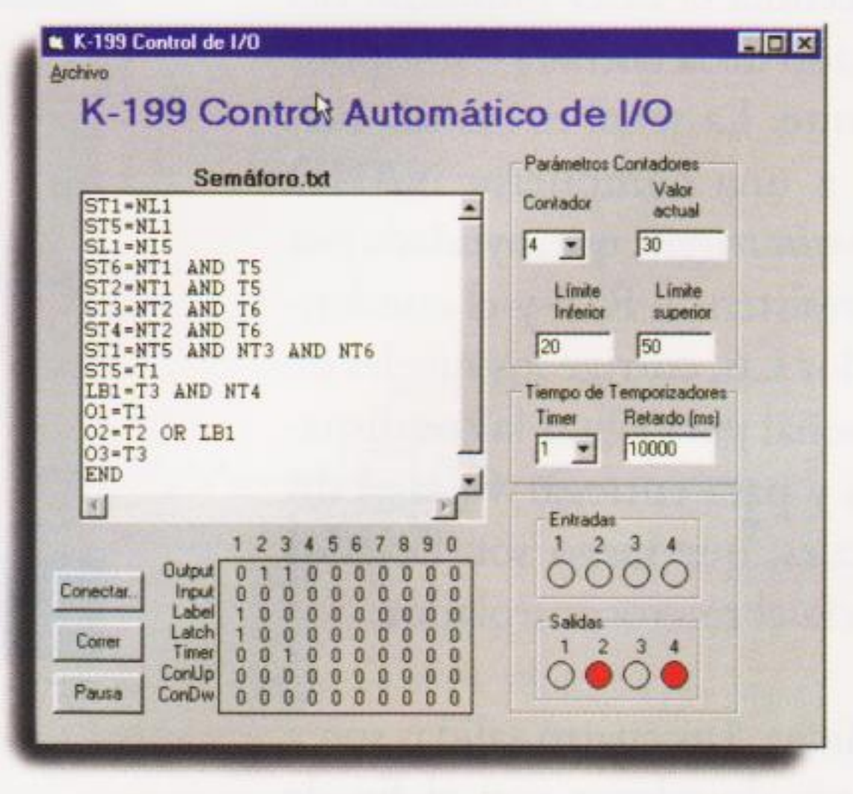


Figura 21.9 *Programación de secuencias automáticas*

las salidas se comporten de una o de otra manera. En la **figura 21.9** puede ver la ventana de edición del programa. Dicha ventana tiene dos botones para ejecutar o detener el programa que se encuentra en la sección de edición.

Más adelante veremos como escribir un programa en la ventana de edición. Cuando haya finalizado la escritura del programa, éste podrá ser almacenado en el disco duro de la PC (o en cualquier otra unidad de almacenamiento), para que sea llamado posteriormente por el programa que lo ejecuta en modo **Run**.

Modo de ejecución (Run). Luego de establecer la comunicación con el dispositivo electrónico de control, deberá cargarse una secuencia previamente archivada en la computadora durante el proceso de ***Edición***. Una vez que sea llamado uno de los programas, inmediatamente empezarán a ejecutarse las órdenes que éste suministre luego del procesamiento automático. En este programa, **figura 21.10**, no se pueden ver los estados de las variables ni se puede modificar el contenido de las instrucciones. Para ello es necesario volver al modo de ***Edición***.

Figura 21.10 *Ejecución de un programa de secuencia automática*

Programación de las secuencias automáticas

Las variables representan un estado digital durante el desarrollo de cualquier programa de secuencia automática. Cada variable posee un índice que la identifica, por ejemplo I1 señala la entrada número 1 y T8 el temporizador número 8.

De igual manera, a cada variable se le puede asignar un valor de acuerdo a determinadas operaciones booleanas. Veamos primero cuales son estas operaciones, y luego veremos las variables y la forma de utilizarlas.

Operaciones

Las operaciones básicas que se pueden utilizar en la programación son:

- **AND**: equivale a la misma operación AND de la lógica booleana, donde el resultado será 1 si **ambas** variables son iguales a 1. Ejemplo:

  O4= I2 AND I3. La salida cuatro será 1 si las entradas dos y tres valen 1, si no, el resultado será 0.

- **OR**: Equivale a la misma operación OR de la lógica booleana, donde el resultado será 1 si cualquiera de las dos variables vale 1. Ejemplo:

  O4= I2 OR I3. La salida cuatro será 1 si cualquiera de las entradas dos y tres vale 1, si ambas son 0, el resultado será 0.

- **NOT (N)**: equivale a la misma operación NOT de la lógica booleana, donde el resultado será el estado contrario al de la variable. Ejemplo:

  O4= N I2. La salida cuatro es igual a la entrada dos negada, si dicha entrada vale 0, el resultado será 1 y viceversa.

Variables

Conozcamos las diferentes variables permitidas en la programación y la forma de utilizarlas. Recordemos que cada una de ellas va acompañada de un índice que la identifica, el cual va de 1 a 10. También se debe tener en cuenta que algunas variables van relacionadas con ciertas instrucciones y con ciertos parámetros de configuración.

- **Entradas (I).** Desde I1 hasta I4 representan el estado de las cuatro entradas de la tarjeta electrónica. Las demás (I5..I10), aunque no representan ninguna entrada física, pueden utilizarse como posiciones de memoria.

Instrucción	I
Parámetros	Ninguno

Ejemplos:

O2 = I4. La salida dos es igual al estado de la entrada cuatro, **figura 21.11**.

O3 = I4 AND N I2. La salida tres es igual a la operación AND entre la entrada cuatro y la entrada dos negada.

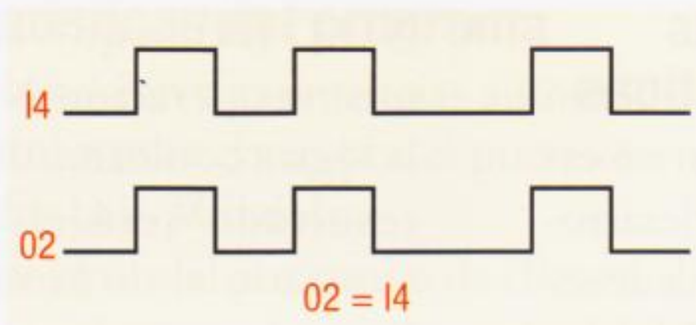


Figura 21.11 *Funcionamiento de las instrucciones de entradas y salidas*

- **Salidas (O).** Desde O1 hasta O4, representan el estado de las cuatro salidas, es decir, si O2 es igual a 1, el relevo número dos estará activado; pero si vale 0, el relevo estará desactivado. Las demás salidas (O5..O10) pueden usarse como posiciones de memoria.

Instrucción	O
Parámetros	Ninguno

Ejemplos:
O2 = I4 . La salida tres es igual al estado de la entrada uno.
O2 = I1 OR N O1. La salida dos es igual a la operación OR entre el estado de la entrada uno y el estado de la salida uno negada.

- **Cerrojos (L).** Los cerrojos (*latchs*) son posiciones de memoria que cuando se activan (estado = 1), permanecen así hasta que otra condición diferente a la de activación las desactive. Se comportan de igual manera a los cerrojos utilizados en electrónica digital. Se tienen 10 cerrojos enumerados desde L1 hasta L10.

Instrucciones	L	SL	RL
Parámetros	Ninguno		

Veamos las instrucciones para activar y desactivar:
Set latch (SL): activa el cerrojo (estado=1). Permanece en 1 hasta que se cumpla una instrucción de *reset latch*.

Reset latch (RL). Desactiva el cerrojo (estado=0). Permanece en 0 hasta que una condición de *set latch* se cumpla.

Ejemplos (ver **figura 21.12**):
SL4 = I3. El cerrojo cuatro se activa cuando la entrada tres sea uno.
RL4 = I2. El cerrojo cuatro se desactiva cuando la entrada dos sea uno.
O3=L4. La salida tres es igual al estado del cerrojo cuatro.

- **Temporizadores(T).** Los temporizadores (*timers*) también son posiciones de memoria que dependen de un lapso de tiempo determinado. Por ejemplo, si se activa T1 y está programado para 5 segundos, su valor será 1 mientras transcurra dicho tiempo, y será 0 cuando finalicen los 5 segundos.

En total se han establecido 10 módulos temporizadores, los cuales reciben la denominación desde T1 hasta T10, donde cada uno de ellos tiene un tiempo de programación en milisegundos, siendo su máximo valor 65536 (65 segundos).

Instrucciones	T	ST	RT
Parámetros	Tiempo (ms)		

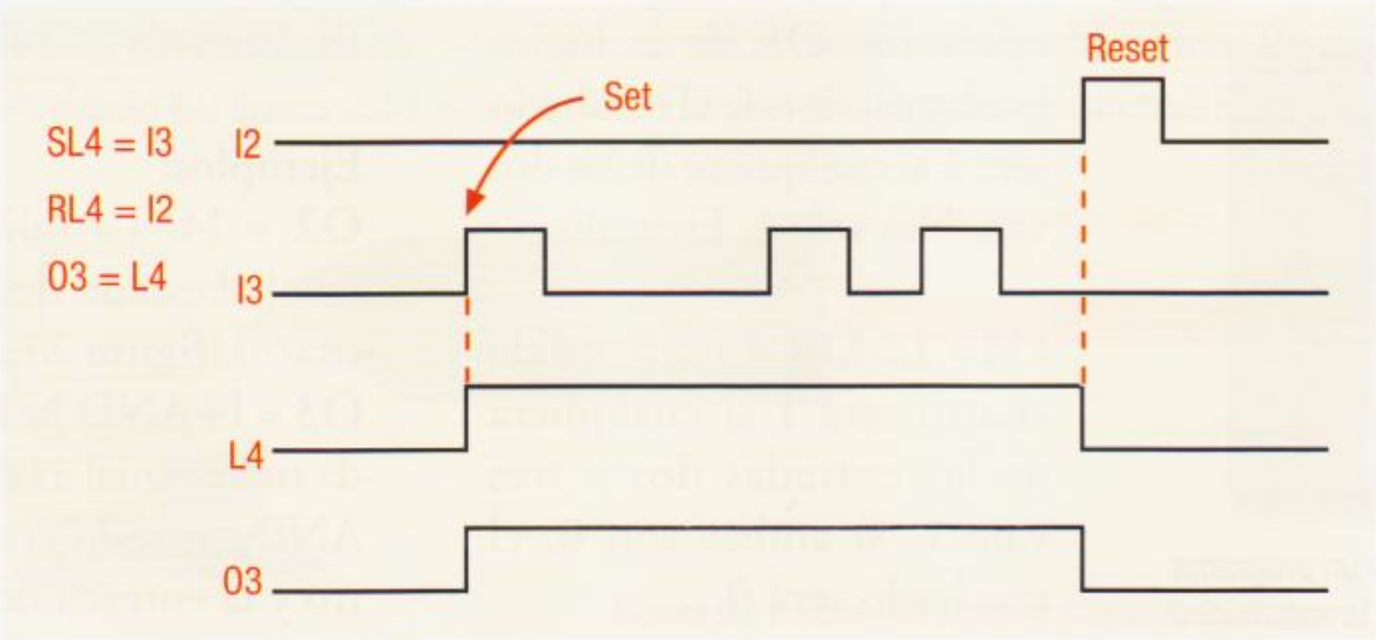


Figura 21.12 *Comportamiento de las instrucciones relacionadas con el cerrojo*

Poseen dos instrucciones para activar y desactivar.

Set timer (ST): activa el temporizador (estado=1). Su valor será 1 hasta que se cumpla el tiempo programado o hasta que se de una instrucción de *reset timer*.

Reset timer (RT): desactiva el temporizador (estado=0). Obliga a finalizar el tiempo del temporizador volviendo su estado a 0. Si el temporizador no está activo, esta instrucción no tiene efecto.

Ejemplos: ver **figura 21.13.**
Si el temporizador (*timer*) cero está programado para 5 segundos:
ST4 = I3. El *timer* cuatro se activa cuando el estado la entrada tres sea 1. Empieza la cuenta del tiempo.
RT4 = I2. El *timer* cuatro se desactiva cuando la entrada dos sea 1.
O3=T4. La salida tres es igual al estado del *timer* cuatro.

- **Etiquetas (LB).** Las etiquetas (*labels*), son posiciones de memoria que se usan para facilitar las operaciones con otras variables del programa. También son diez etiquetas, desde LB1 hasta LB10.

Instrucciones	LB
Parámetros	Ninguno

Ejemplos con etiquetas:
LB6= I4 AND I3. La etiqueta 6 es igual a la operación AND entre la entrada cuatro y la entrada tres.
O2=LB6. La salida dos es igual al estado de la etiqueta 6.

- **Contadores (CU y CD).** Los contadores se utilizan para contabilizar pulsos en las entradas digitales o sucesos con las demás variables. Son posiciones de memoria que valen 0 ó 1, de acuerdo al valor de una cuenta. Las cuentas pueden ser ascendentes o descendentes y se efectúan con los flancos de subida de la señal involucrada.

Instrucciones	Parámetros
C	Límite superior del contador
CU	Límite inferior del contador
CD	Valor actual del contador
RC	

Cada contador tiene dos variables diferentes que pueden estar en 1 ó en 0:

CU (*Counter up*): es el estado del contador con referencia al límite superior del mismo. Por ejemplo, si el límite superior del contador número dos es 50 y el valor actual está en 49, el estado de CU2 es 0. Pero si el valor actual es 51, el estado de CU2 será 1 ya que es mayor al límite superior.

CD (*Counter down*): es el estado del contador con referencia al límite inferior del mismo. Por ejemplo, si el límite inferior del contador número dos es 30 y el valor actual está en 25, el estado de CU2 es 1 ya que es menor al límite inferior. Pero si el valor actual es 31, el estado de CU2 será 1.

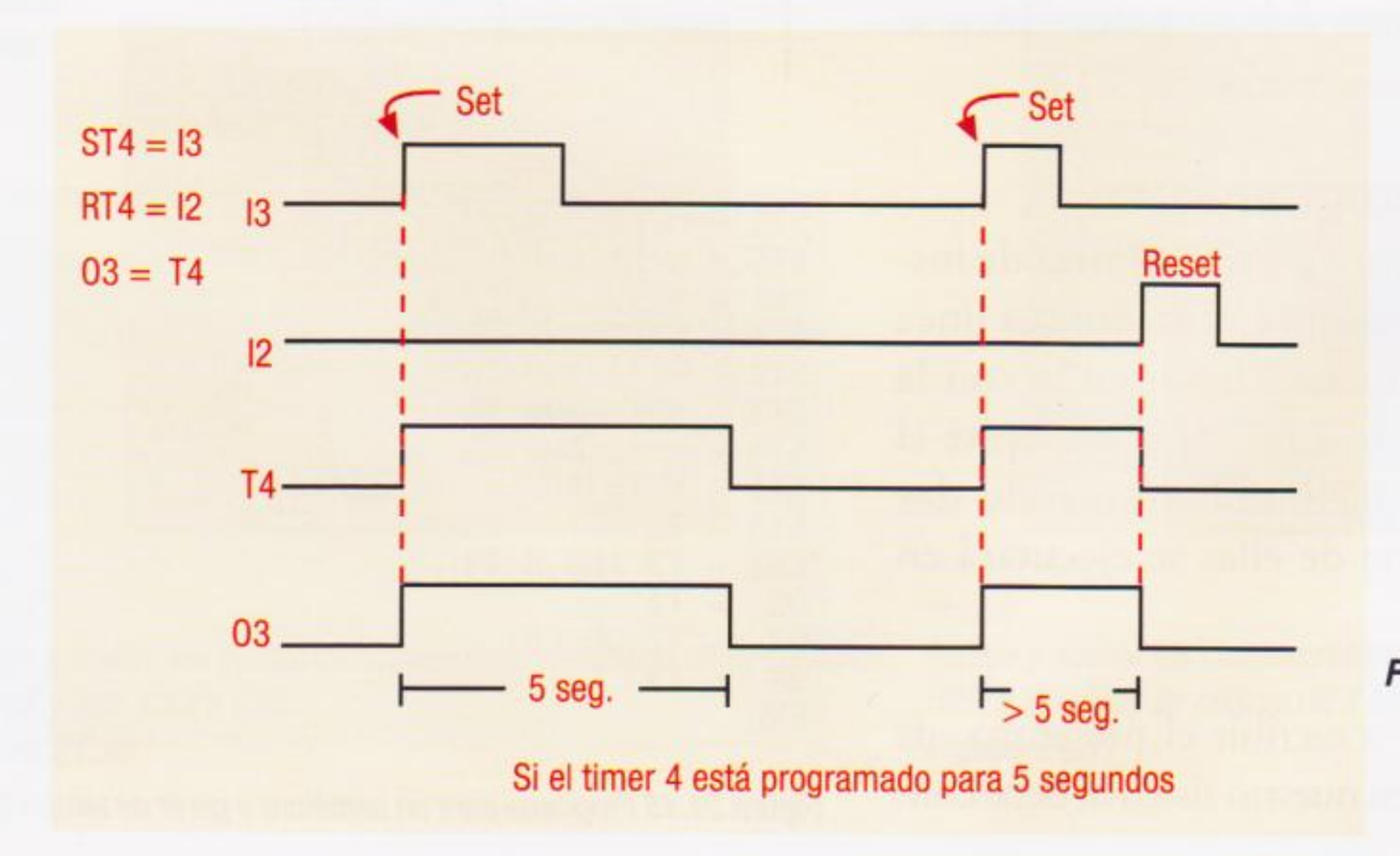


Figura 21.13 *Comportamiento de las instrucciones relacionadas con los temporizadores*

También existe una instrucción que hace referencia a los contadores, se trata del ***reset counter***:

RC (***Reset counter***). Esta instrucción hace que el valor actual del contador se inicie en cero. Es utilizada especialmente luego de que se ha cumplido un límite superior o un límite inferior.

Ejemplos con contadores:

CU2= I4. Incrementa el contador 2 cada vez que la entrada cuatro pasa de 0 a 1. Si el valor actual de C2 es 25, cuando la entrada cuatro pase de 0 a 1, el valor actual de C2 será 26.

CD2= I3. Decrementa el contador dos cada vez que la entrada tres pasa de 0 a 1. Si el valor actual de C2 es 25, cuando la entrada tres pase de 0 a 1, el valor actual de C2 será 24.

O2= CU5. La salida dos es igual al estado del contador 5 con respecto al límite superior. Si el valor actual de C5 es 50 y el límite superior del mismo está configurado en 40, el estado de CU5 será 1.

O2= CD5. La salida dos es igual al estado del contador 5 con respecto al límite inferior. Si el valor actual de C5 es 50 y el límite inferior del mismo está configurado en 51, el estado de CD5 será 1.

RC5= I2. El valor actual del contador cinco se iniciará en 0 cuando la entrada dos sea 1.

Estructura de un programa

Un programa está hecho a partir de líneas de instrucciones. La computadora ejecuta línea a línea empezando por la primera y finalizando con la que tenga la instrucción END. Luego repite el ciclo nuevamente. Si una línea tiene más de dos instrucciones, cada una de ellas se ejecutará en forma secuencial.

Antes de empezar a escribir el programa, de determinada secuencia en nuestro sistema, debe configurar los parámetros que tendrán los temporizadores y los contadores, seleccionando los tiempos, los límites superiores y los límites inferiores.

Para ilustrar mejor el funcionamiento de nuestro sistema, a continuación se muestra un programa que emula el funcionamiento de un semáforo de tráfico vehicular.

- **Programa para el control de luces en un semáforo**. Para este programa fueron necesarios cinco temporizadores, los cuales fueron configurados con los siguientes tiempos:

T1: 10000 (10 s)
T2: 2000 (2 s)
T3: 10000 (10 s)
T4: 8000 (8 s)
T5: 12000 (12 s)
T6: 4000 (4 s)

Las tres luces del semáforo se asignaron de la siguiente manera:

Luz verde: salida 1 (O1)
Luz amarilla: salida 2 (O2)
Luz roja: salida 3 (O3)

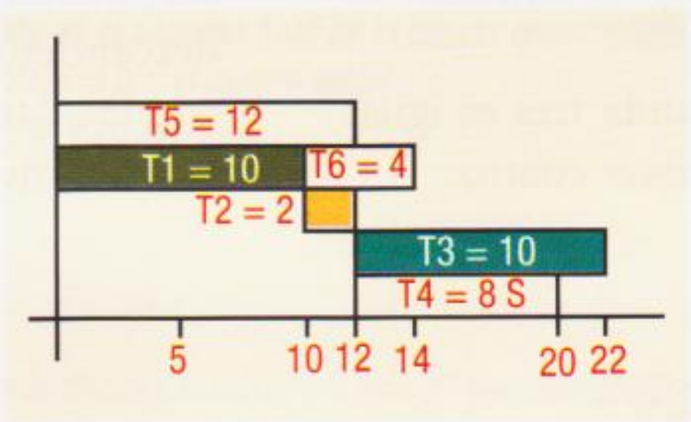


Figura 21.14 *Diagrama de tiempos para el programa del manejo de las luces de un semáforo*

```
ST1 = N L1
ST5 = N L1
SL1 = N I5
ST6 = N T1 AND T5
ST2 = N T1 AND T5
ST3 = N T2 AND T6
ST4 = N T2 AND T6
ST1 = N T5 AND N T3 AND N T6
ST5 = T1
LB1 = T3 AND N T4
O1  = T1
O2  = T2 OR LB1
O3  = T3
END
```

Figura 21.15 *Programa para un semáforo a partir de temporizadores*

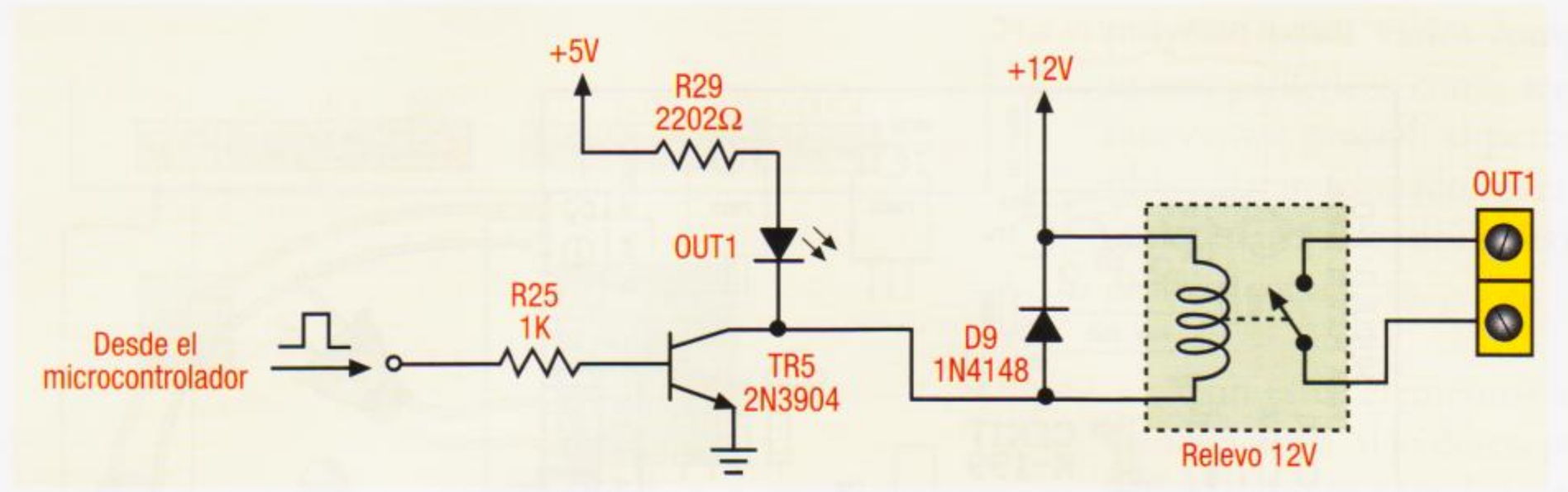


Figura 21.16 *Diagrama esquemático de una de las salidas*

A cada luz se le asigna un tiempo de la siguiente manera:

Luz verde: T1 (10 s)
Luz amarilla: T2 (2 s)
Luz roja: T3 (10 s)

Observe en la **figura 21.14** el diagrama de tiempos utilizado para este ejemplo, y en la **figura 21.15**, el listado del programa. Ensáyelo y tomelo como punto de partida para desarrollar procesos automáticos más complejos.

Ensamblaje de la tarjeta electrónica

Uno de los métodos más utilizados dentro de la electrónica digital para activar circuitos de potencia partiendo de señales digitales, es por medio de transistores y relevos.

La señal del circuito digital (en este caso el microcontrolador) excita la base del transistor quien a su vez hace disparar el relevo. Los contactos del

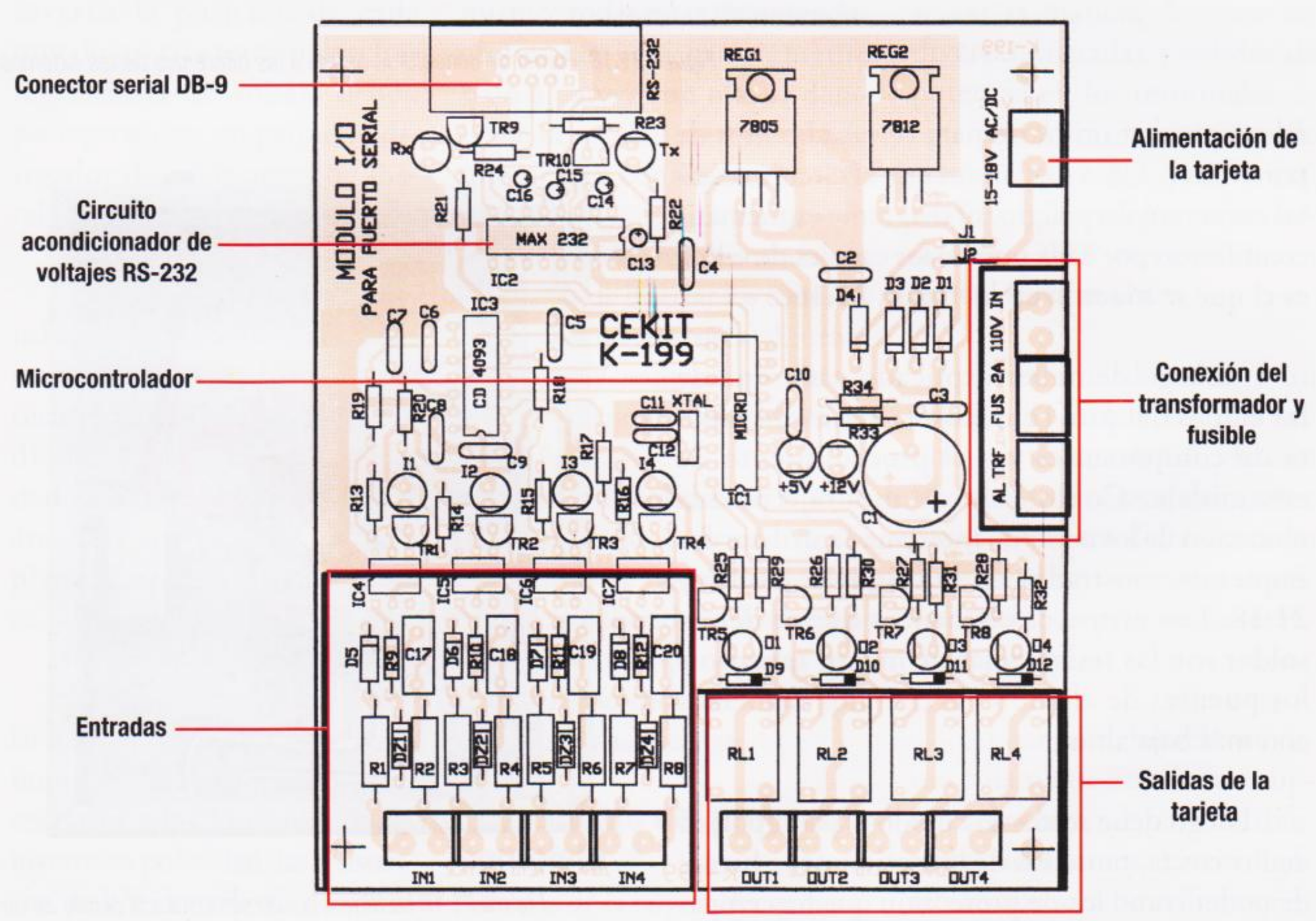


Figura 21.17 *Guía de montaje del proyecto*

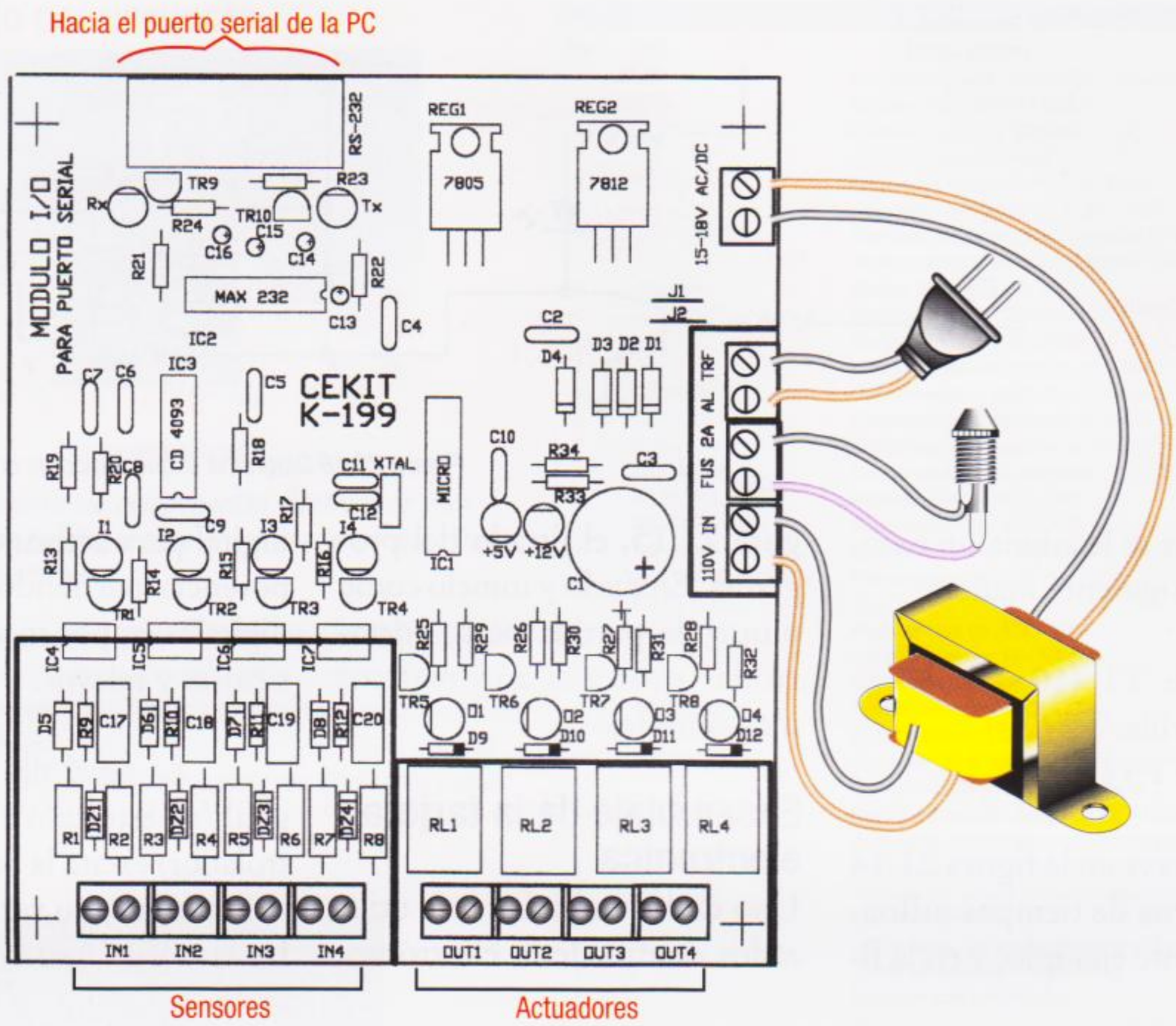


Figura 21.18 *Forma de conectar la tarjeta a las diferentes partes externas*

relevo pueden utilizarse para cerrar circuitos de potencia de CA ó de CC sin que el circuito digital corra ningún peligro. El diagrama esquemático utilizado por cada uno de los canales de salida es el que se muestra en la **figura 21.16**.

Luego de haber explicado cada una de las etapas del proyecto, con la ayuda de la lista de componentes vea el procedimiento de ensamblaje. Como guía de montaje para la ubicación de los componentes puede utilizar los esquemas mostrados en las **figuras 21.17 y 21.18**. Los primeros elementos que se deben soldar son las resistencias de menor tamaño y los puentes de alambre, es decir los elementos con más baja altura.

Luego debe soldar los diodos con la polaridad correcta, tanto los de la sección de la fuente de poder como los de protección que hay en paralelo con las bobinas de los relevos. A continua-

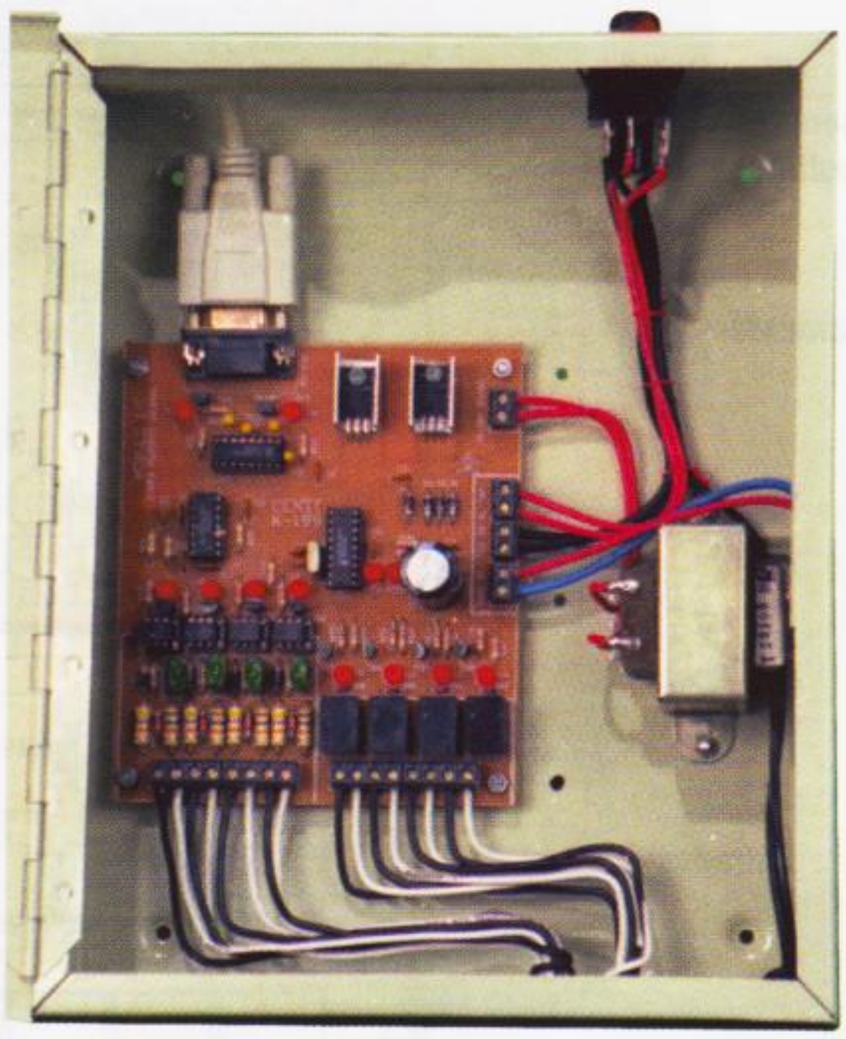


Figura 21.19 *La tarjeta puede ser instalada dentro de un gabinete industrial*

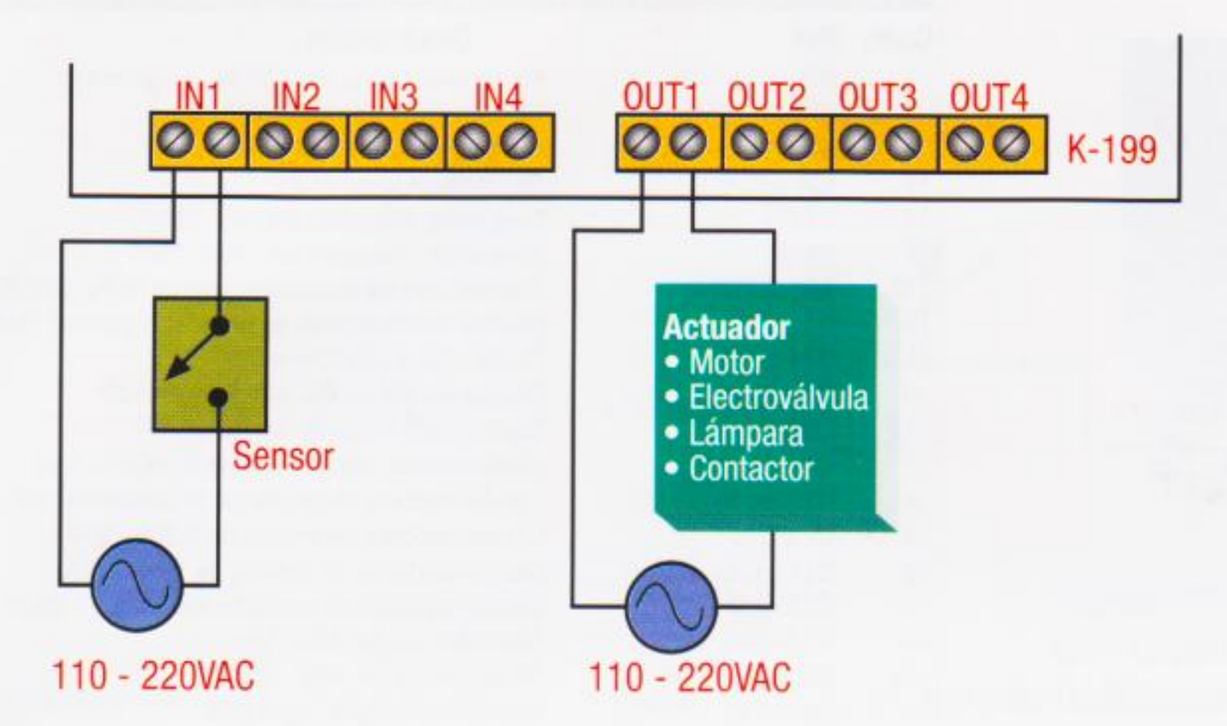


Figura 21.20 *Como conectar un sensor y un actuador*

ción, se instalan las bases de los circuitos integrados teniendo como guía la muesca en uno de sus lados, la cual debe estar orientada hacia el pin 1.

También puede soldar los transistores cuidando de no invertir la posición de cada uno de sus tres terminales. Los reguladores de voltaje deben protegerse con un pequeño disipador de calor antes de instalarlos en la tarjeta.

Los condensadores cerámicos pueden soldarse sin importar la polaridad de sus terminales. Ahora, suelde los diodos LED, revisando cuidadosamente la ubicación del ánodo y del cátodo; el lado plano se orienta hacia la fuente negativa.

Posteriormente, puede instalar los elementos de mayor tamaño, tales como los condensadores polarizados, cuidando de no invertir su polaridad, las resistencias de 1 W, los relevos y los bornes de conexión con tornillo.

También puede soldar el conector DB9 para circuito impreso. Por último, se suelda el cristal de 4 MHz y se insertan los circuitos integrados en sus respectivas bases. Recuerde que la muesca del circuito integrado va hacia el pin número 1 del mismo; todas estas recomendaciones las verá gráficamente paso a paso y con mayor detalle en la sección de ensamblaje que sigue a continuación.

Una buena recomendación es que antes de insertar los circuitos integrados en sus bases, alimente el circuito y verifique que los niveles de voltaje sean los adecuados.

Antes de aplicar corriente al circuito, revise cada una de las soldaduras de tal modo que estén bien adheridas a los terminales de los componentes, y que no estén haciendo contacto con otro punto adyacente.

Ensamblaje del proyecto completo

Además de la tarjeta electrónica, referencia K-199 de CEKIT, el proyecto posee varios componentes periféricos como son el interruptor general, el portafusible y el transformador, para los cuales están a disposición sus respectivos conectores.

Con estos elementos adicionales usted, si lo desea, puede instalar la tarjeta dentro de un gabinete metálico con los respectivos orificios de acceso para el interruptor, la alimentación y el puerto serial, como se muestra en la **figura 21.19**. De esta manera el proyecto toma la apariencia de un dispositivo industrial.

Forma de conectar sensores y actuadores

En la **figura 21.20** puede observar la manera de conectar diversas entradas y salidas de potencia en los terminales de la tarjeta. Para evitar errores de conexión, verifique la conexión de las salidas en los terminales OUT y las entradas en los terminales IN.

Cada entrada admite un voltaje de CA de 110 ó 220 voltios y un voltaje de CC, de 24 a 50 voltios. El sensor que utilice para activar una de las entradas debe suministrar los dos terminales del voltaje, como se muestra en la figura.

Ahora bien, los relevos sirven solamente como interruptores de corriente para los diferentes actuadores. La configuración que se debe utilizar, también aparece en la **figura 21.20**.

Ensamblaje del proyecto

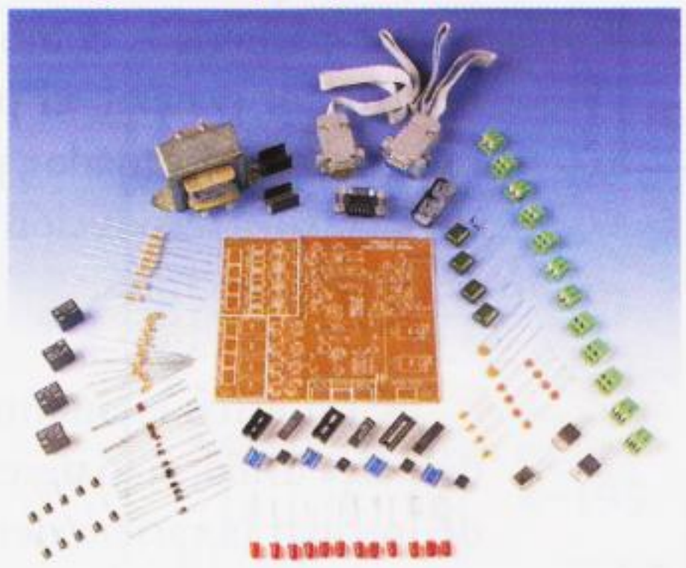
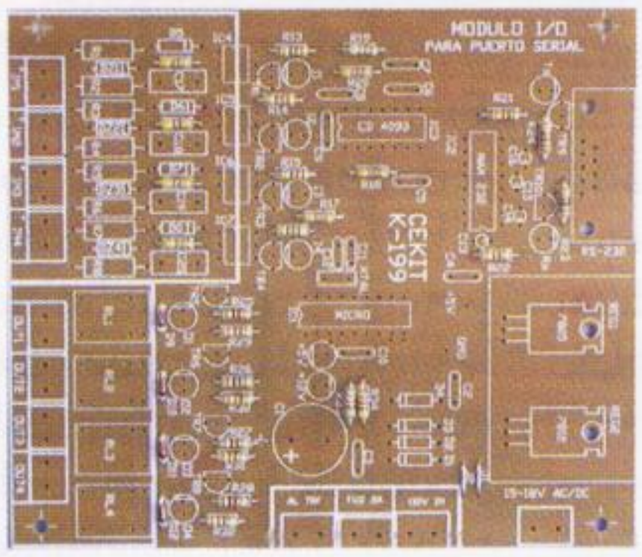


Figura 21.21 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el control I/O de potencia por el puerto serial.*

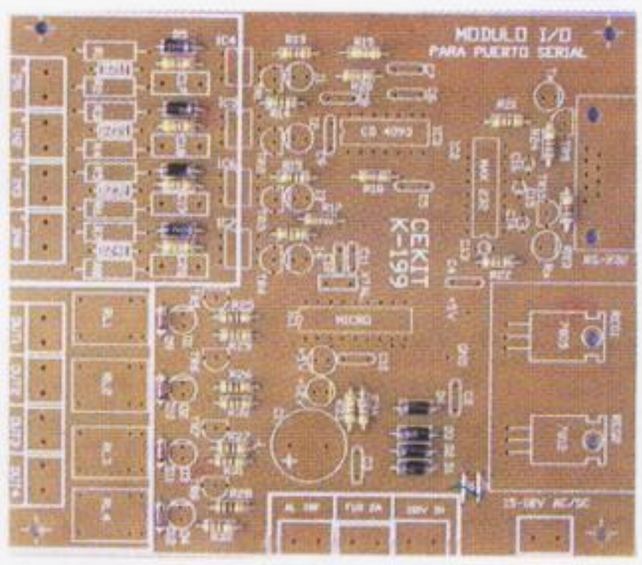
Por medio de este proyecto y con la ayuda de una computadora personal se podrán controlar muchos dispositivos eléctricos y electrónicos, enviando y recibiendo las señales correspondientes de control a través de un puerto serial RS232 de la PC. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

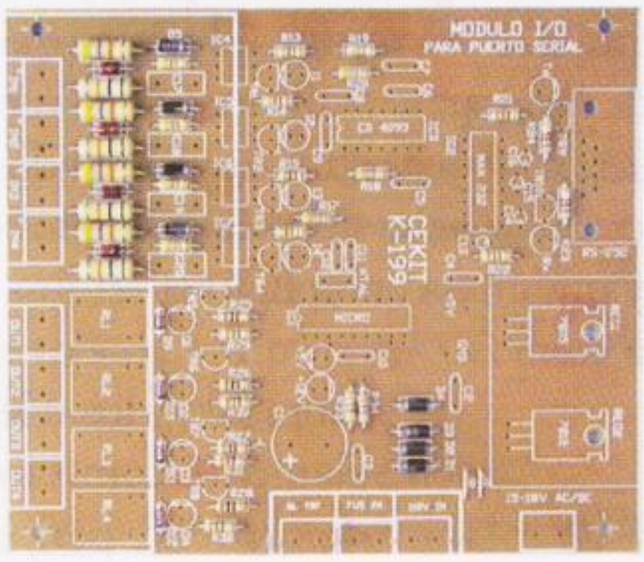
Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F84, programado
1	IC2	Circuito integrado MAX232
1	IC3	Circuito integrado CD4093
4	IC4 a IC7	Optoacopladores 4N25
1	REG1	Regulador integrado de +5V, 7805
1	REG2	Regulador integrado de +12V, 7805
10	TR1 a TR10	Transistores de propósito general NPN, 2N3904
8	D1 a D8	Diodos rectificadores de propósito general 1N4004
4	DZ1 a DZ4	Diodos *Zener* de 12V a 1W
4	D9 a D12	Diodos rectificadores rápidos 1N4148
12	LEDS	Diodos LED rojos de 5mm
1	C1	Condensador electrolítico de 2.200uF - 35V
4	C13 a C16	Condensadores electrolíticos de tantalio de 1uF - 35V
8	C2 a C10	Condensadores cerámicos de 0,1uF - 50V
2	C11 y C12	Condensadores cerámicos de 22pF - 50V
4	C17 a C20	Condensadores de poliéster de 0,22uF - 250V
4	R1,R3,R5 y R7	Resistencias de 47k - 1W
4	R2,R4,R6 y R8	Resistencias de 10k - 1W
4	R9 a R12	Resistencias de 4,7k - 1/4W
12	R13 a R16, R21,R22 R29 a R34	Resistencias de 220 - 1/4W
4	R17 a R20	Resistencias de 47k - 1/4W
6	R23 a R28	Resistencias de 1k - 1/4W
1	XTAL	Cristal de cuarzo de 4 MHz
4	RL1 a RL4	Relés miniatura de 12VCC
1	K-199	Circuito impreso CEKIT
1	RS232	Conector DB-9 hembra para circuito impreso
12		Conectores de tornillo de 2 pines
1		Base para circuito integrado de 18 pines
1		Base para circuito integrado de 16 pines
1		Base para circuito integrado de 14 pines
4		Bases para circuito integrado de 8 pines
1		Transformador M508
1		Cable serial uno a uno DB-9M a DB-9H
1		Cable de potencia con enchufe
1		Disquete con programas
2		Tornillos de 1/4 X 1/8 con sus tuercas
2		Disipadores de calor para TO-220



Paso 1
Instale y suelde los puentes de alambre, las resistencias de 1/4W, y los diodos rápidos 1N4148, D9 a D12.
Figura 21.22

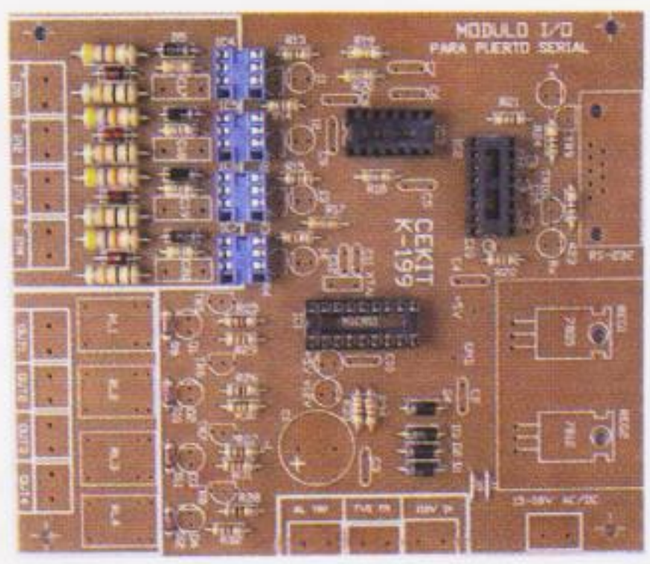


Paso 2
Luego instale y suelde los diodos rectificadores de propósito general D1 a D8 de 1A, con referencia 1N4004.
Figura 21.23



Paso 3
Instale y suelde los diodos Zener de 5,1V D1 a D4, y las resistencias de 1W de 10kΩ y 47kΩ.
Figura 21.24

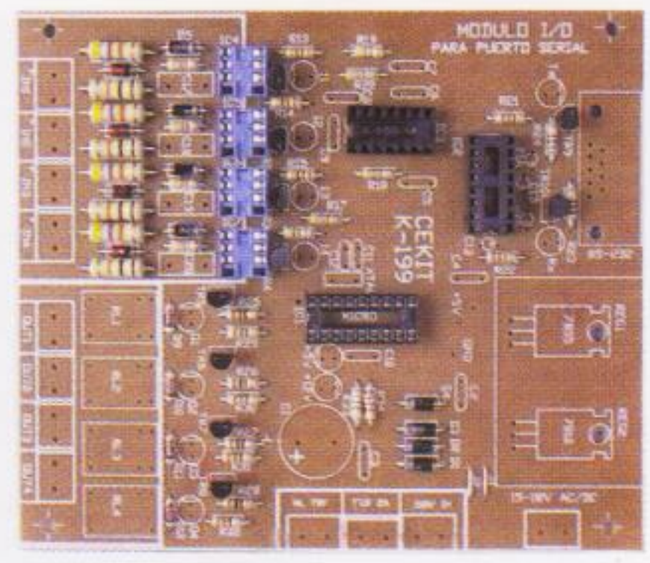
Use unas pinzas y extraiga con ellas los pines 4 y 5 de las cuatro bases de 8 pines, para formar las bases de 6 pines requeridas por los optoacopladores.
Figura 21.25



Paso 4

Instale y suelde las bases para los circuitos integrados IC1 a IC7, respetando la orientación indicada en la guía de ensamblaje.

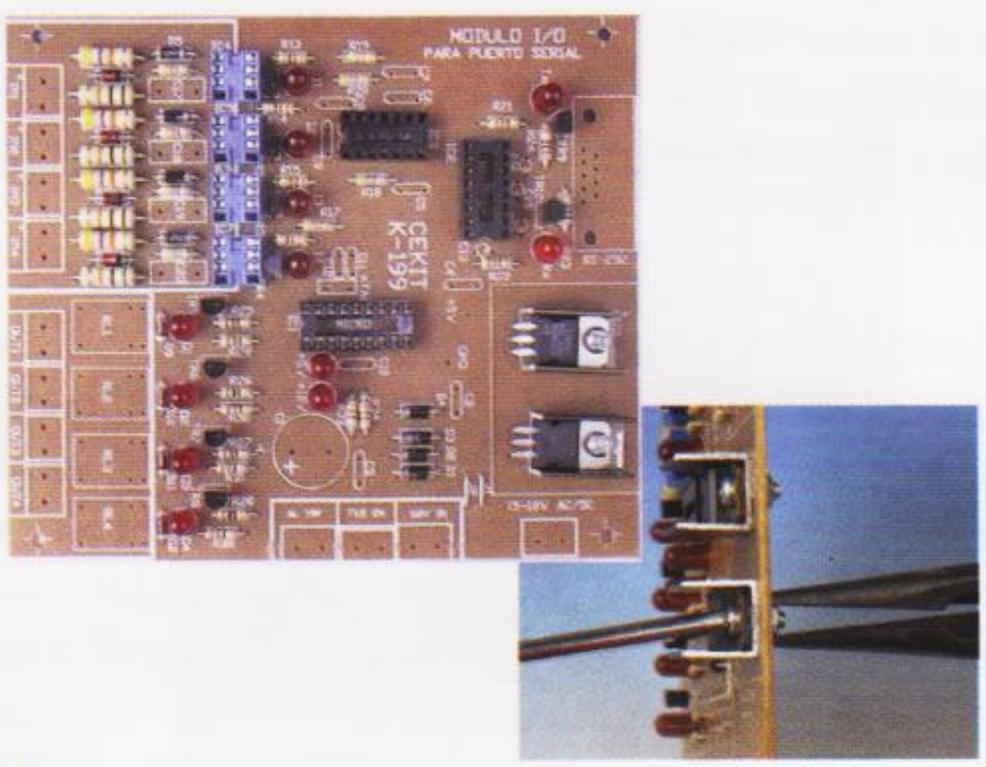
Figura 21.26



Paso 5

Instale y suelde los transistores de propósito general NPN TR1 a TR10.

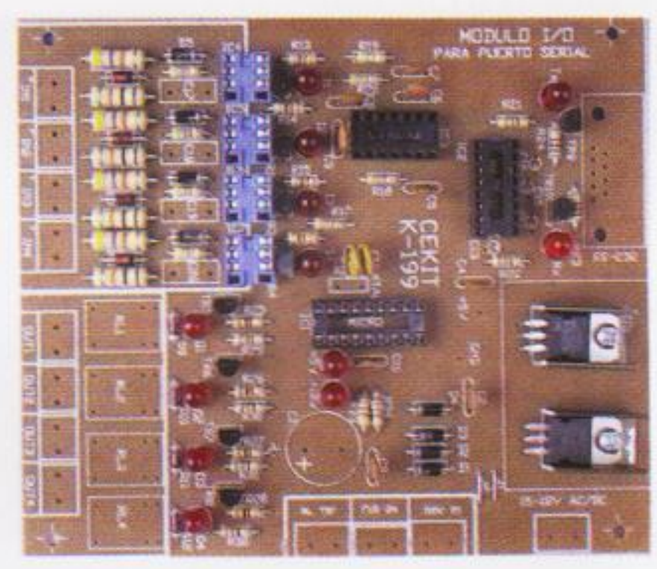
Figura 21.27



Paso 6

Instale y suelde los diodos LED de entrada y salida e indicadores de comunicación serial, y asegure firmemente, con tornillos de 1/4" por 1/8" , los dispadores de calor de los dos reguladores de voltaje en el espacio indicado por la guía de ensamblaje.

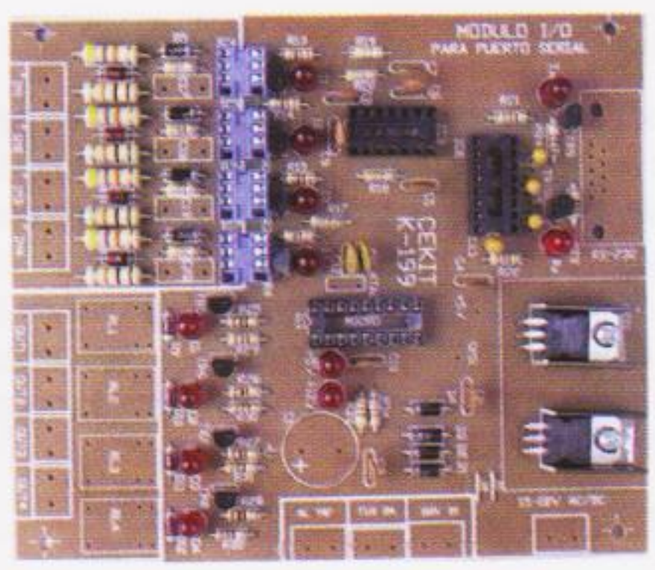
Figura 21.28



Paso 7

Instale y suelde todos los condensadores cerámicos de 50V C2 a C12.

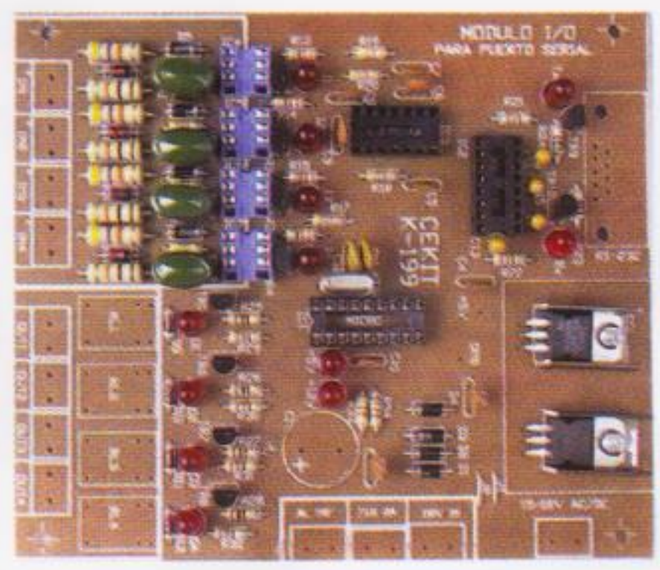
Figura 21.29



Paso 8

Instale y suelde los cuatro condensadores electrolíticos de tantálio de 1µF y 35V, C13 a C16.

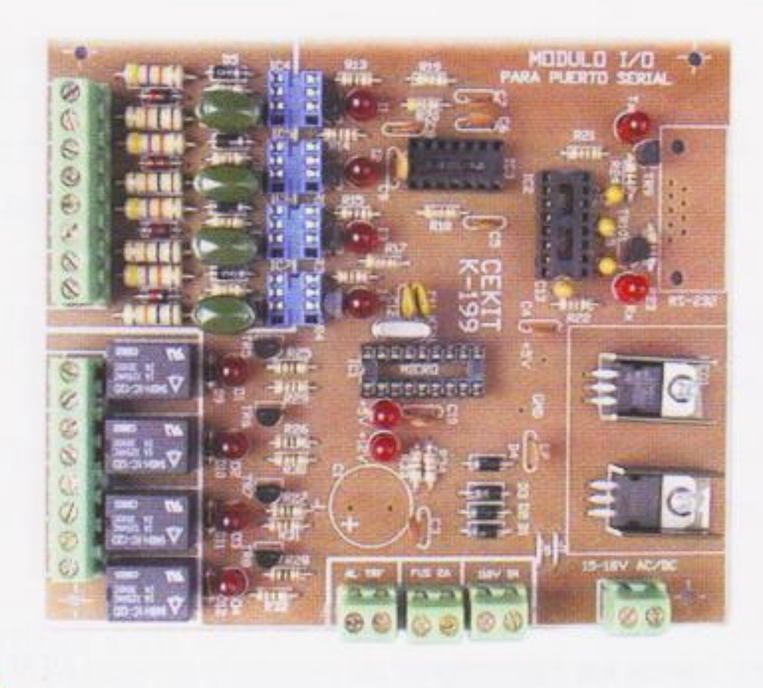
Figura 21.30



Paso 9

Instale y suelde los condensadores de poliéster de 250V C17 a C20, y el cristal de cuarzo de 4 MHz XTAL.

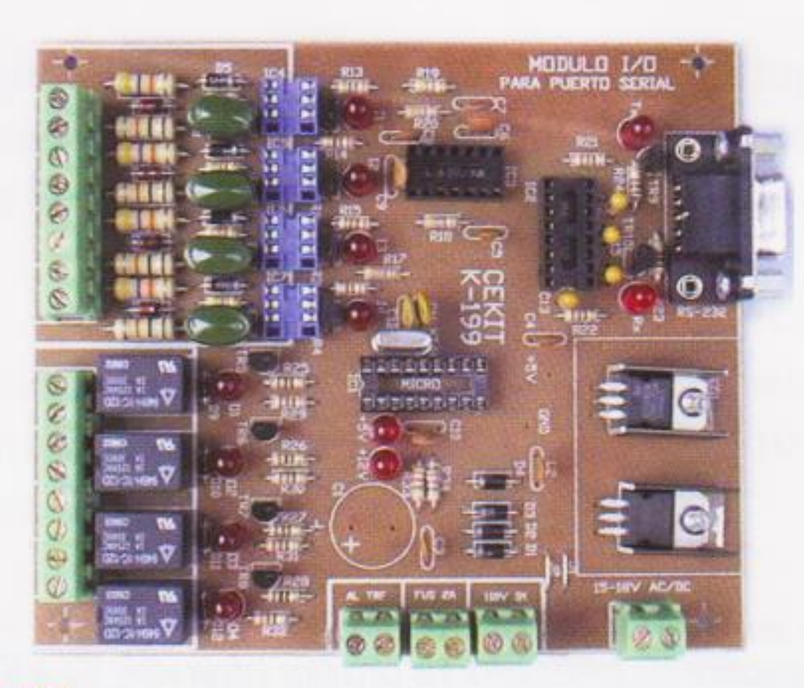
Figura 21.31



Paso 10

Instale y suelde los cuatro relés miniatura de 12V en el espacio indicado para las salidas, y los doce conectores de tornillo para las entradas y salidas de la tarjeta.

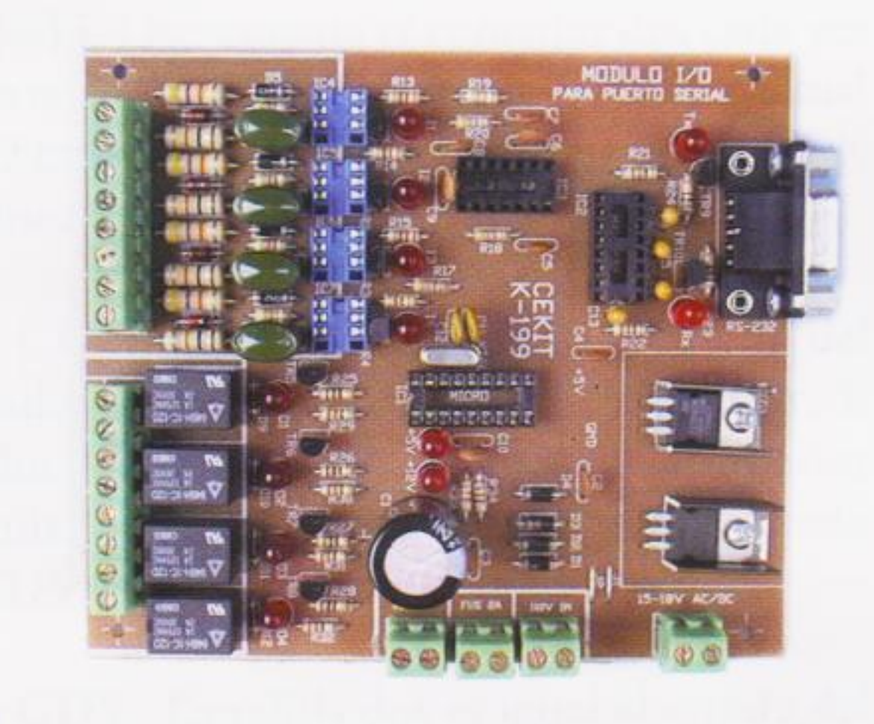
Figura 21.32



Paso 11

Instale y suelde el conector hembra DB-9 para el puerto de comunicaciones serial RS232

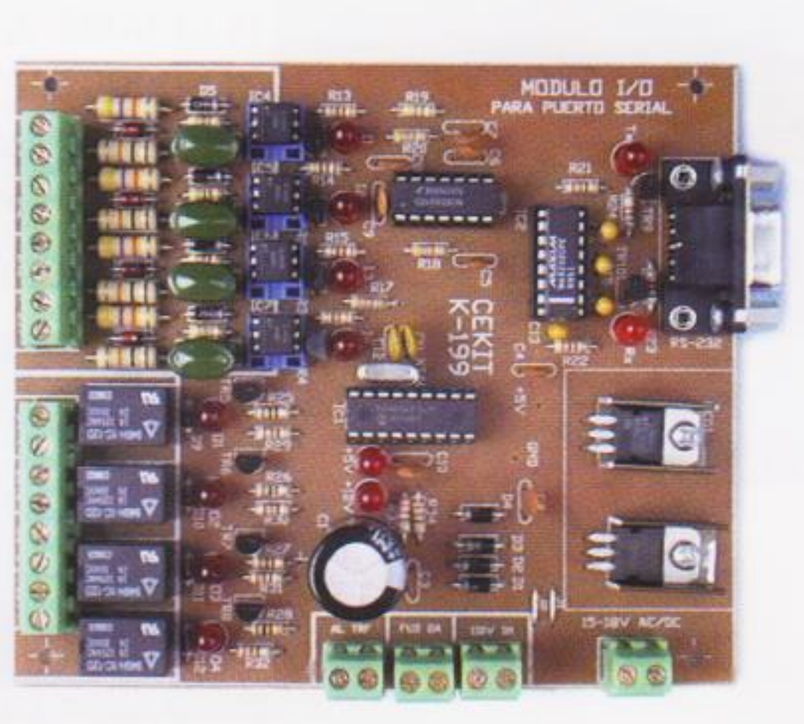
Figura 21.33



Paso 12

Por último, instale y suelde el condensador electrolítico C1 de 2.200 µF y 50V.

Figura 21.34



Paso 13

Éste es el aspecto final de la tarjeta ensamblada del control I/O de potencia por el puerto serial.

Figura 21.35

En la figura se muestra la tarjeta del control I/O de potencia por el puerto serial controlando el movimiento de un brazo robotizado, a partir de las señales enviadas desde una computadora por dicho puerto. De la tarjeta se han utilizado algunas de sus salidas para manejar los motores de las articulaciones, y también algunas entradas para recibir la posición real indicada por los microinterruptores del robot.

Figura 21.35

Paso 14. Pruebas finales

Para alimentar el circuito utilice el conector tipo tornillo ubicado en el lado derecho de la tarjeta, tal como se mostró la guía de ensamblaje; cuando está correctamente alimentado, los LED indicadores de +5V y +12V deben encenderse. Conecte el puerto serial de la computadora al conector DB-9 del circuito impreso y ejecute uno de los programas incluidos en el disco; por medio de los LED Tx y Rx, verifique la transferecia de los datos.

PROYECTO 22

ALARMA PARA MOTOCICLETA

Construya este proyecto y adquiera un sencillo pero efectivo sistema de seguridad para su sistema de transporte. Soluciones como ésta confirman que los microcontroladores nos pueden hacer la vida mejor.

Características

- Facilidad de manejo
- Microcontrolador Motorola
- Salida por medio de relevo
- Tamaño reducido, lo que facilita su instalación
- Diodo LED indicador del estado de la alarma

En esta oportunidad desarrollaremos un sistema sencillo de alarma para motocicleta basado en un microcontrolador Motorola, el cual, por su tamaño reducido, puede ser ocultado fácilmente dentro del *chasis*.

Funcionamiento

La activación de la alarma se hace por medio de la llave de encendido de la motocicleta. Si, estando el motor apagado, usted da vuelta a la llave de encendido (ON) y dentro de un período de tiempo no mayor a diez segundos cierra el encendido (OFF), el sistema se arma o se pone en estado de vigilancia.

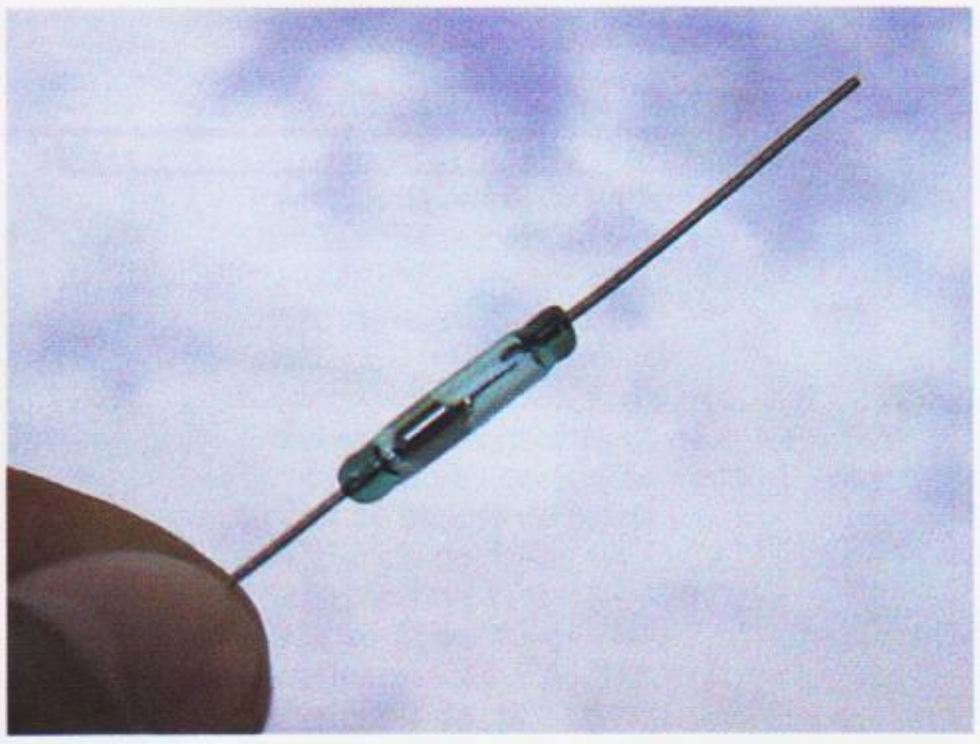


Figura 22.1 *Sensor de movimiento*

Desde el momento en que se activa el encendido y durante dichos diez segundos, el LED indicador permanecerá titilando para indicar que se encuentra en esta etapa del proceso. Si en este espacio de tiempo el circuito detectó la desactivación del encendido el LED se apagará, indicando que el sistema está listo y disponible para detectar cualquier movimiento. En este estado el microcontrolador chequea permanentemente el estado de un sensor de movimiento para originar una señal de alarma, cuando este sensor sea activado.

Sensor de movimiento

El sensor de movimiento utilizado es un sensor de mercurio común, **figura 22.1**, que se puede encontrar fácilmente en el mercado. Este sensor consiste básicamente de un contacto normalmente abierto, el cual se cierra cuando, debido al movimiento del sensor, el mercurio líquido se acerca al contacto y sirve de puente entre los dos puntos. Este sensor debe estar colocado en un lugar estratégico de la motocicleta y como ya lo mencionamos, será chequeado permanentemente una vez armado el sistema.

Cuando desarrolle algún proyecto de esta naturaleza, lo más importante que se debe garantizar es que el circuito eléctrico esencial de la motocicleta quede independiente de la nueva conexión de la alarma. Esto quiere decir que se deben evitar problemas tales como que la mo-

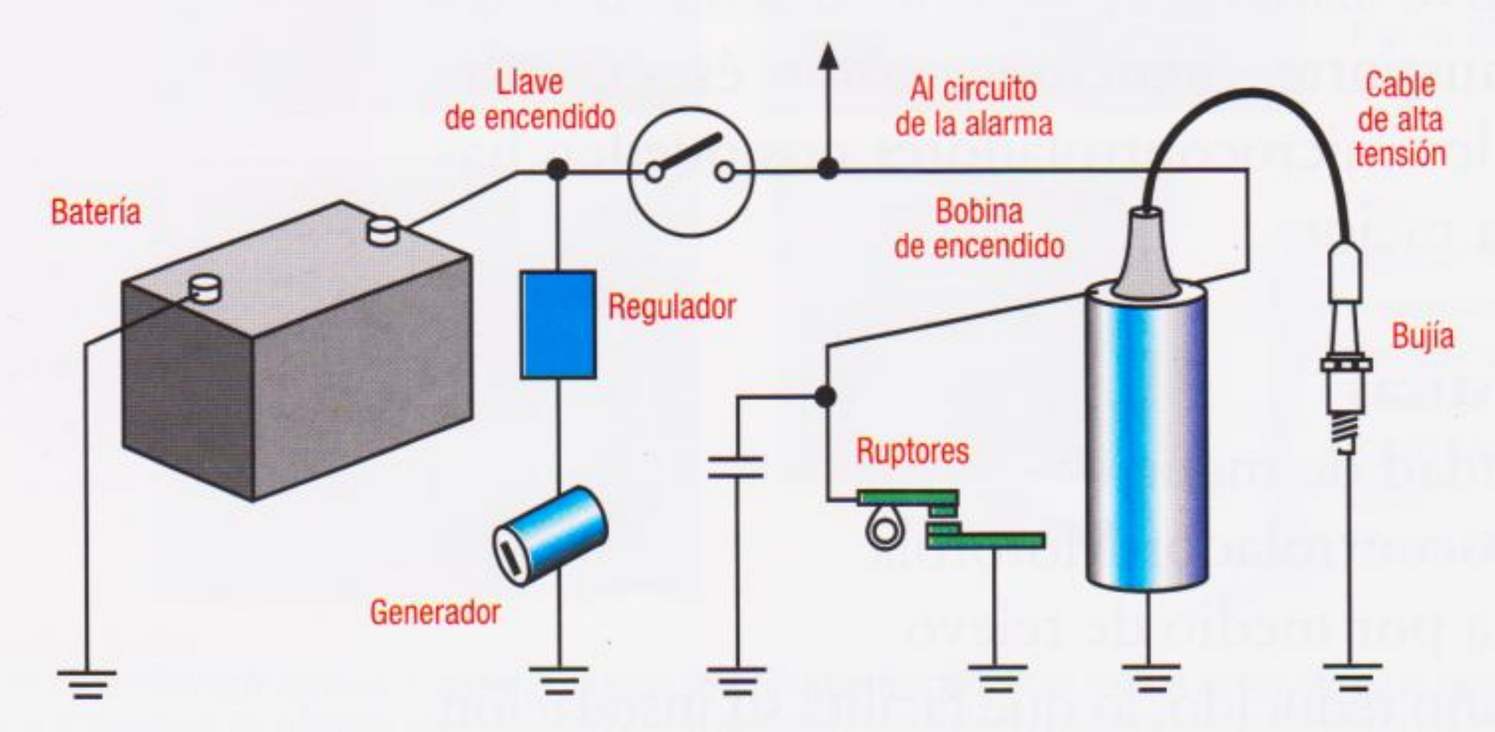


Figura 22.2 *Esquema básico del encendido de una motocicleta*

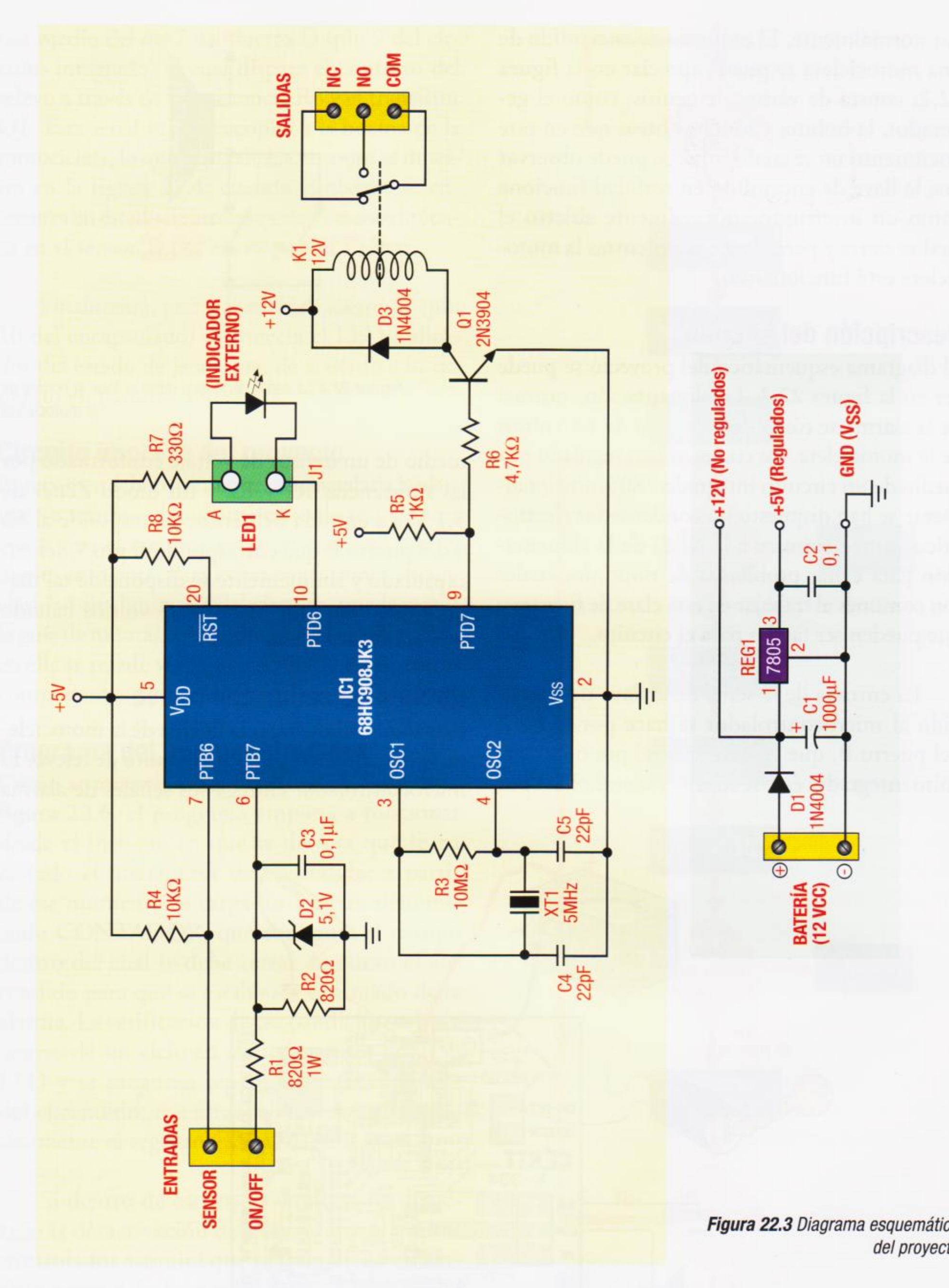


Figura 22.3 *Diagrama esquemático del proyecto*

tocicleta no funcione en el caso de que el circuito de la alarma se averíe o se humedezca. Lo ideal es que en caso de emergencia, el circuito pueda ser «deshabilitado» por cualquier persona que no tenga conocimiento alguno en electrónica y la motocicleta pueda funcio-

nar normalmente. El esquema de encendido de una motocicleta se puede apreciar en la **figura 22.2**; consta de varios elementos, como el generador, la bobina y muchos otros que en este documento no se analizarán. Se puede observar que la llave de encendido en realidad funciona como un interruptor normalmente abierto el cual se cierra y permanece así mientras la motocicleta esté funcionando.

Descripción del circuito

El diagrama esquemático del proyecto se puede ver en la **figura 22.3**. La alimentación normal de la alarma se toma de la batería de 12 Voltios de la motocicleta, los cuales se han regulado por medio de un circuito integrado 7805, adicionalmente se han dispuesto un condensador electrolítico y uno cerámico a la salida de la alimentación para evitar problemas de ruido, los cuales son comunes al trabajar en esta clase de fuentes y que pueden ser fatales para el circuito.

La entrada de la señal de la llave de encendido al microcontrolador se hace por el *bit* 7 del puerto B, que corresponde al pin 6 del circuito integrado; éste se rebaja o acondiciona por medio de un divisor de voltaje conformado por las resistencias R1 y R2 y un diodo Zener de 5,1 V. La entrada del sensor corresponde al *bit* 6 del puerto B que corresponde al pin 7 del encapsulado y simplemente se dispone de tal manera que conmute entre 0 y 5 voltios cuando está activado o desactivado.

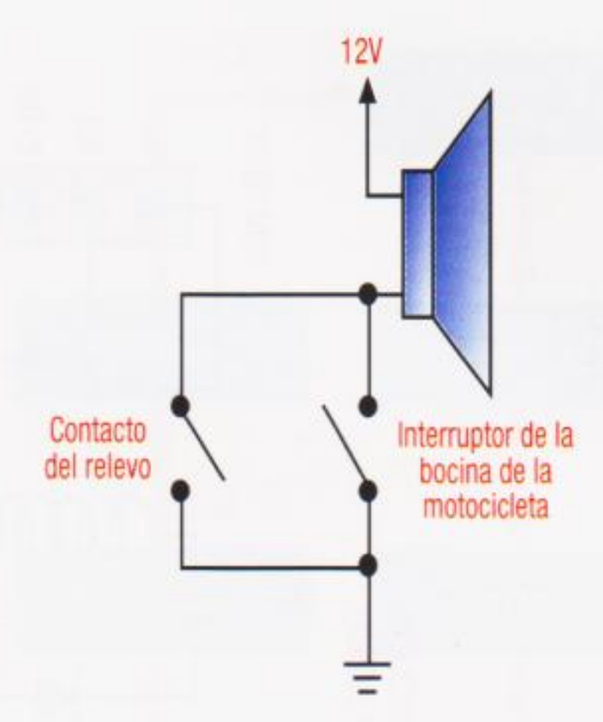


Figura 22.4 *La señal del relevo debe activar la bocina de la motocicleta*

Circuito de salida con relevo

Para dar la salida hacia la bocina de la motocicleta, se decidió incorporar un circuito de relevo. El microcontrolador entrega sus señales de alarma

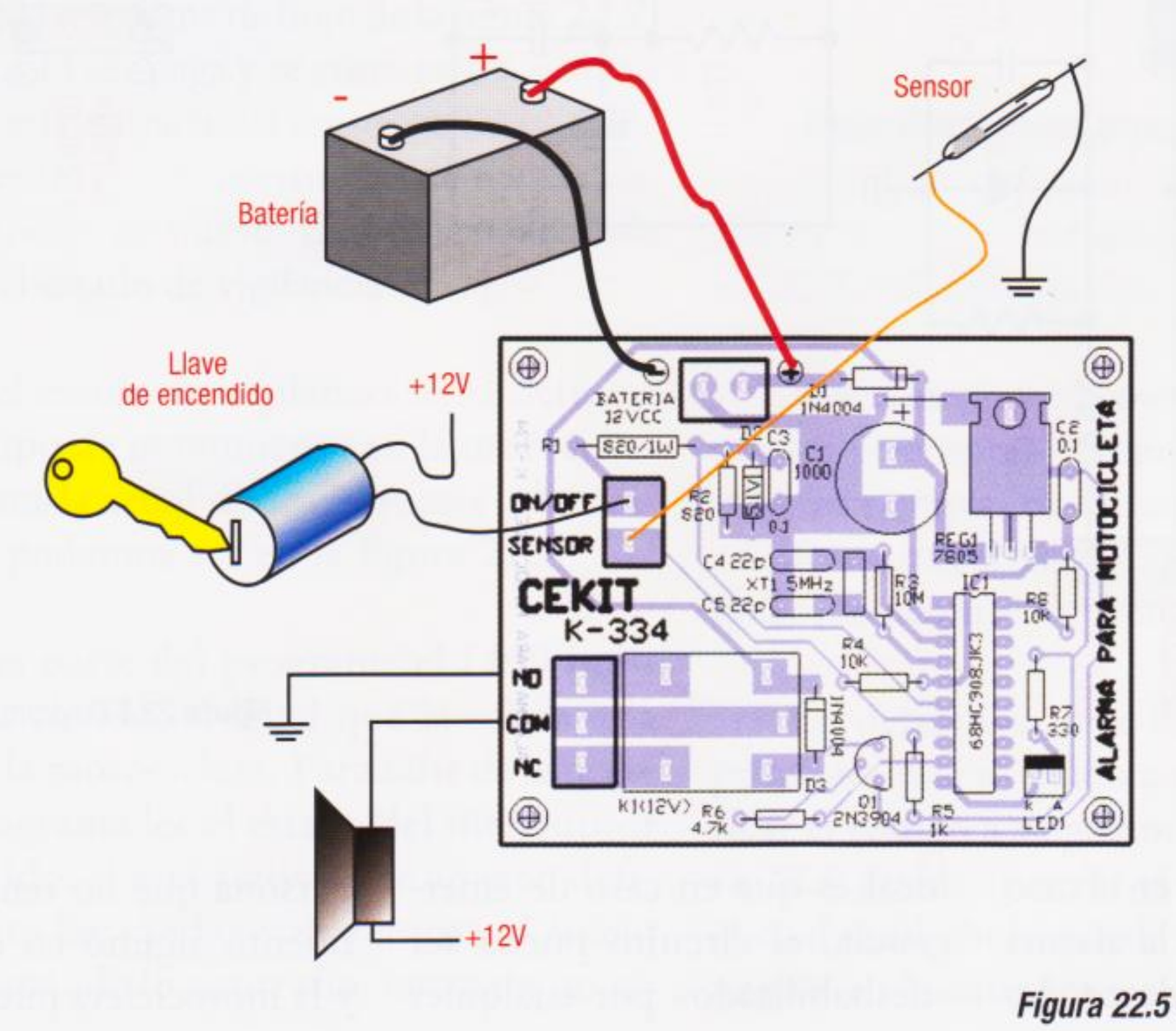


Figura 22.5 *Guía de montaje de la alarma*

por medio del *bit* 7 del puerto D (pin 9 del circuito integrado) el cual dispara el contacto del relevo a través de la resistencia R6 y el transistor Q1. Esta señal se debe acoplar a la bocina de la motocicleta, lo que se puede hacer como se muestra en la **figura 22.4**; cuando la alarma se encuentra en estado de armado y detecta movimiento en el sensor, la bocina empezará a sonar.

Finalmente, por el *bit* 6 del puerto D (pin 10 del encapsulado) se conecta el LED indicador del estado de la alarma, de acuerdo a lo expuesto en párrafos anteriores.

Circuito impreso del proyecto

Es recomendable que una vez sea instalada la alarma, se recubra con alguna clase de resina, para evitar que el circuito entre en contacto con el medio ambiente. Para evitar confusiones y tener claridad en el montaje del circuito, puede seguir la guía de montaje que se ilustra en la **figura 22.5**, en ella se puede ver la ubicación de los distintos componentes dentro del circuito.

Programa del microcontrolador

Como se muestra en el diagrama de flujo de la **figura 22.6**, el programa empieza a funcionar desde el instante en que se detecta que se ha cerrado el interruptor de encendido; a partir de ese momento se carga un registro denominado CONTADOR, que configura el tiempo dentro del cual se debe cerrar de nuevo el encendido para que se establezca el armado de la alarma. La verificación de esa condición se hace dentro de un ciclo en el cual se hace titilar el LED y se pregunta repetitivamente el estado del encendido, mientras se decrementa paulatinamente el registro CONTADOR.

Si dentro de ese intervalo no se ha detectado la desactivación del encendido, el microcontrolador asumirá que se trata de un encendido normal de la motocicleta.

Si por el contrario se detecta la señal de desactivación del encendido, el programa salta a la

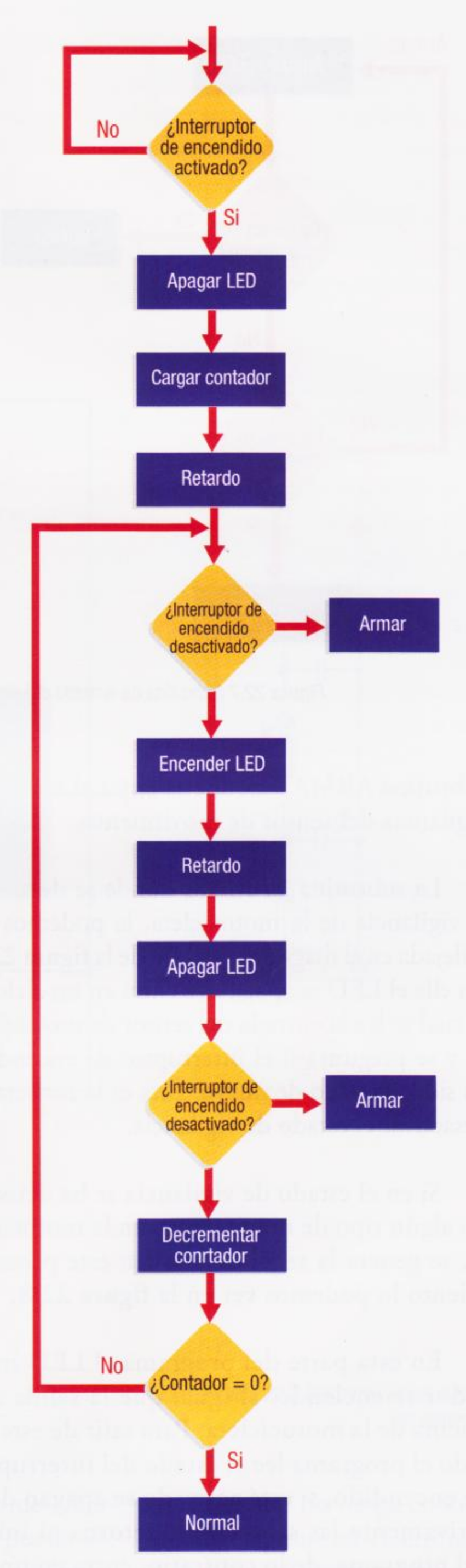


Figura 22.6 *Cuerpo principal del programa*

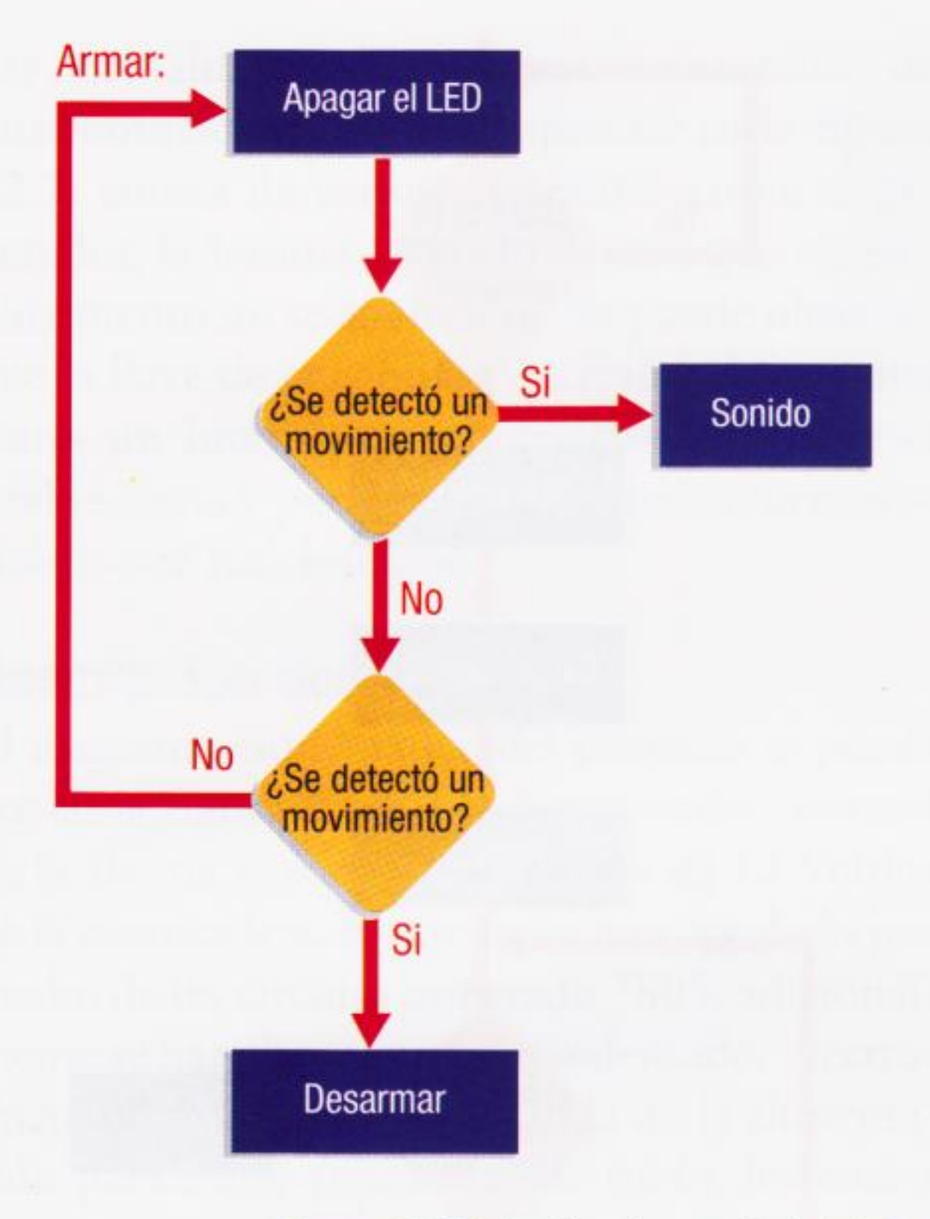


Figura 22.7 *Subrutina de armado del sistema*

Figura 22.8 *Subrutina de generación de la señal de alarma*

subrutina ARMAR donde se entra al estado de vigilancia del sensor de movimiento.

La subrutina ARMAR, donde se desarrolla la vigilancia de la motocicleta, la podemos ver reflejada en el diagrama de flujo de la **figura 22.7**. En ella el LED se apaga y se entra en un ciclo en el cual se lee la entrada del sensor de movimiento y se pregunta si el interruptor de encendido ha sido activado de nuevo. Ésta es la manera de desactivar el estado de vigilancia.

Si en el estado de vigilancia se ha detectado algún tipo de movimiento en la motocicleta, se genera la señal de alarma; este procedimiento lo podemos ver en la **figura 22.8**.

En esta parte del programa el LED indicador se enciende, al igual que la salida a la bocina de la motocicleta. Para salir de este estado el programa lee el estado del interruptor de encendido, si está activado se apagan definitivamente las salidas y se retorna al inicio del programa, de lo contrario, entra en un ciclo en el que la bocina permanece activada y el LED titila de nuevo.

Seguramente se le pueden agregarle muchas otras características de funcionamiento a esta alarma, por ejemplo, puede adicionarle un interruptor secreto con el cual se pueda desactivar en cualquier momento el funcionamiento del motor.

En definitiva, si le llama la atención este proyecto, está invitado a profundizar y modificar el código a su gusto; para ello, en la **figura 22.9** puede apreciar el programa completo, el cual también lo puede descargar de nuestra página en la internet.

```
************************
*
* Software en assembler para el HC08 JK3,
* PROYECTO: ALARMA PARA MOTOCICLETA
* --
* Juan Andres Castaño Welgos
* welgos@cekit.com.co
*
* PEREIRA
* COLOMBIA-2001
* PTB7: MOTOCICLETA ENCENDIDA
* PTB6: SENSOR MAGNETICO
************************
$include 'jl3regsg.inc'

RAMStart      EQU  $0080    ; DIRECCIÓN ORIGEN RAM
RomStart      EQU  $EC00    ; DIR. ORIGEN DE LA ROM
Vectores      EQU  $FFDE
RESET_VEC     EQU  $FFFE

      org RAMStart

CONTADOR        RMB     1  ; VARIABLES UTILIZADAS
REGIS1          RMB     1  ; DENTRO DEL PROGRAMA.
REGIS2          RMB     1  ; UBICACIÓN EN LA
CONTA1          RMB     1  ; MEMORIA RAM.
BANDERA         RMB     1
      org RomStart

*****************************************************
*  - DESDE ESTE PUNTO SE INICA LA EJECUCION DEL
*                    CODIGO
*             DESPUES DE UN RESET.
*
*****************************************************
INICIO:
       RSP
       Bset COPD,CONFIG1   ;DESHABILITA EL COP
                           ;(watchdog)
       CLRA
       CLRX
       MOV #$ff,DDRD       ; CONFIGURO EL PUERTO D
                           ; COMO SALIDAS

       MOV #$000,DDRB      ; CONFIGURO EL PUERTO B
                           ; COMO ENTRADAS

INI:

       CLR      PORTD
       BSET     6,PORTD    ; INICIAMOS CON EL LED OFF
       RSP                 ; INICIANDO EL STACK POIN.
       BRCLR    0,BANDERA,ESPERA1 ; SE CHEQUEA SI SE
       ; ACTIVADO EL BIT 0 DE BANDERA. ESTO SIGNIFICA
       ; QUE LA MOTOCICLETA SE ENCUENTRA TRABAJANDO
       ; EN MODO NORMAL.
       BRSET    7,PORTB,INI
ESPERA1:
       BSET     6,PORTD    ; APAGAR EL LED
       BRCLR    7,PORTB,ESPERA1 ;ESPERA QUE SE
                           ; ENCIENDA LA MOTOCICLETA
                           ; PARA INICIAR EL PROGRAMA
                           ; AL ACTIVARSE, SE
                           ; ENCIENDE EL LED
       BSET     4,PORTD

       ;AHORA DEBE ESPERAR QUE SE APAGUE LA
       ;MOTOCICLETA DENTRO DE UN PERIODO DETERMINADO
ESPERA2:
       ;Y SE HACE PARPADEAR EL LED
       MOV      #$08,CONTA1; 100 VECES .1 SEG. = 10
SEG
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01

ESPERA3:
       RSP
       BRCLR    7,PORTB,ARMA ; SE CHEQUEA EL ENCENDIDO
                             ; PARA DETERMINAR SI SE
                             ; ARMA EL SISTEMA
       BSET     6,PORTD      ; APAGA EL LED
       JSR      RETA01       ; RETARDO DE 0,5 SEGUNDOS
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01
       BCLR     6,PORTD      ; ENCIENDO EL LED
       BRCLR    7,PORTB,ARMA ; LEO DE NUEVO EL
       JSR      RETA01       ; ENCENDIDO
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01
       DBNZ     CONTA1,ESPERA3 ; DECREMENTO CONTA1
       BRA      NORMAL       ; PARA CONFIGURAR EL
                             ;TIEMPO DE ESPERA

ARMA:  JSR      RETA01
       BSET     6,PORTD      ;APAGO EL LED
       BRSET    6,PORTB,SOUND  ; ¿SE MOVIO LA
                             ;MOTOCICLETA?
                             ; EL SISTEMA
       BRSET    7,PORTB,DESARMA ; SI SE ACCIONÓ LA
                             ;LLAVE SE DESARMA
       BRA      ARMA

SOUND  BSET     7,PORTD      ; ACTIVAR LA ALARMA
       BCLR     6,PORTD
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01
       BRSET    7,PORTB,DESARMA ; SI SE ACCIONÓ LA
                             ;LLAVE SE DESARMA
       BCLR     7,PORTD
       BCLR     6,PORTD
       JSR      RETA01
       JSR      RETA01
       JSR      RETA01
       BRA      SOUND
DESARMA:
       BCLR     7,PORTD      ; APAGO LA SALIDA DEL RELÉ
       BSET     6,PORTD      ; APAGO EL LED
;      BSET     0,BANDERA
       BRA      INI          ; INICIO DE NUEVO EL PROCESO

*****************************************************
* RETARDO- SUBRUTINA DE RETARDO POR SOFTWARE *
*                                            *
*****************************************************

RETA01

       LDA      #128T ; RETARDO DE 0.1 s
ET2    CLRX
ET3    DECX
       BNE      ET3
       DECA
       BNE      ET2
       RTS
RET20S
       LDA      #20T
       STA      REGIS1
ET1    JSR      RETA01
       DEC      REGIS1
       BNE      ET1
       RTS

NORMAL:
       NOP
       BSET     0,BANDERA
       Org RESET_VEC
       Dw INICIO    ;al darse reset salta a Start

       END
```

Figura 22.9 *Código fuente del proyecto*

Ensamblaje del proyecto

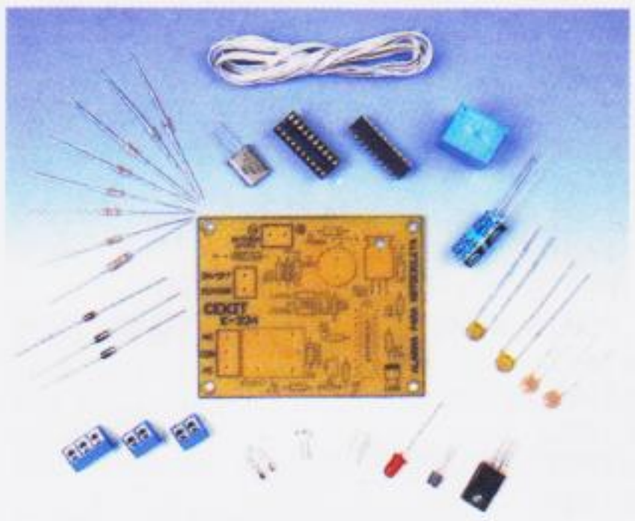


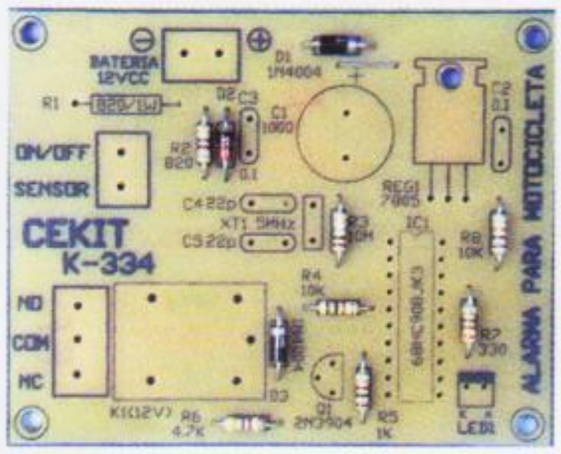
Figura 22.10 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar la alarma para motocicleta K-334.*

Utilizando este circuito usted puede proteger eficazmente su motocicleta contra robo, para ello este proyecto cuenta con una entrada de propósito general para el sensor, y un relé de salida con capacidad de 10A. Con éste se puede interrumpir el sistema de encendido. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

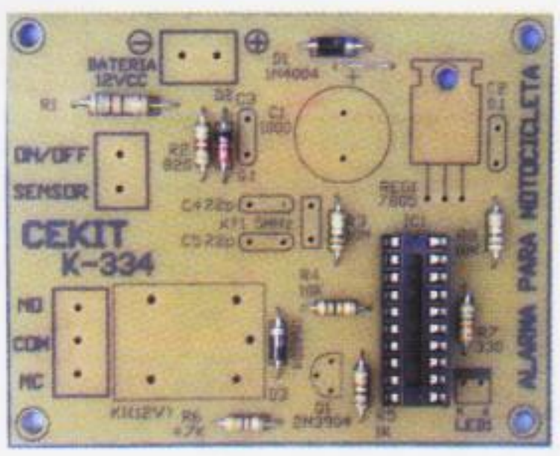
LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador 68HC908JK3, programado
1	REG1	Regulador fijo de +5V -1A, 7805
1	Q1	Transistor de propósito general NPN, 2N3904
2	D1 y D3	Diodos rectificadores de propósito general 1N4004
1	D2	Diodo *zener* de 5.1V a 1W
1	LED1	Diodo LED rojo de 5mm
1	C1	Condensador electrolítico de 1000uF - 25V
2	C2 y C3	Condensadores cerámicos de 0.1uF - 50V
2	C4 y C5	Condensadores cerámicos de 22pF - 50V
1	XT1	Cristal de quarzo de 5MHz
1	R1	Resistencia de 820 - 1W
1	R2	Resistencia de 820 - 1/4W
1	R3	Resistencia de 10M - 1/4W
2	R4 y R8	Resistencias de 10k - 1/4W
1	R5	Resistencia de 1k - 1/4W
1	R6	Resistencia de 4.7k - 1/4W
1	R7	Resistencia de 330 - 1/4W
1	K1	Relé de 12VCC
1	K-334	Circuito impreso CEKIT
1	J1	Conector en línea de 2 pines M/H
1		Conector de tornillo de 3 pines
2		Conectores de tornillo de 2 pines
1		Base para circuito integrado de 20 pines
1m		Cable *ribbon* de 2 hilos

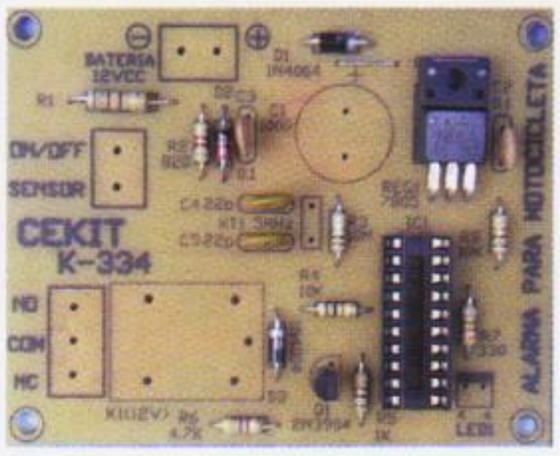
Paso 1
Instale y suelde el puente de alambre, las resistencias de 1/4W, los diodos rectificadores 1N4004, y el diodo Zener de 5,1V.
Figura 22.11

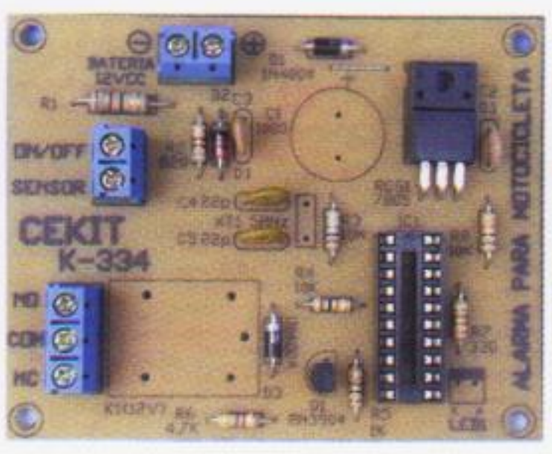


Paso 2
Luego instale y suelde la base para el microcontrolador, y la resistencia R1 de 1W.
Figura 22.12

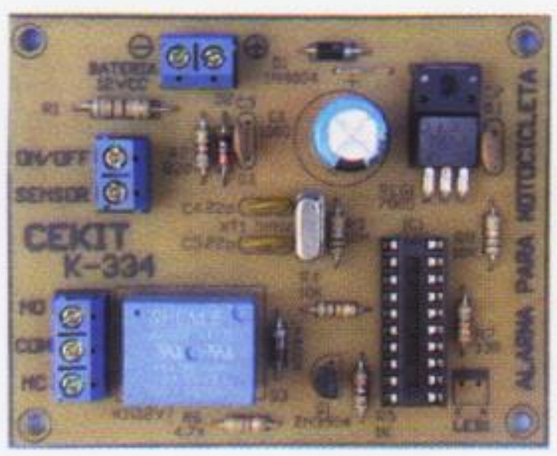


Paso 3
Instale y suelde el regulador integrado de voltaje de +5V, el transistor NPN Q1 y los condensadores cerámicos.
Figura 22.13

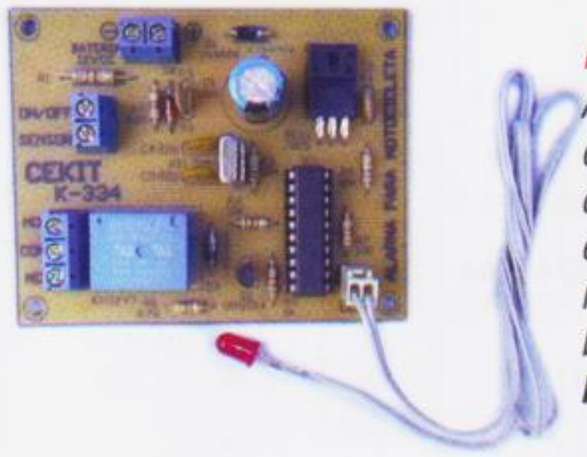




Paso 4
Instale y suelde los conectores de tornillo de dos pines para la alimentación y para la entrada y la salida del circuito.
Figura 22.14



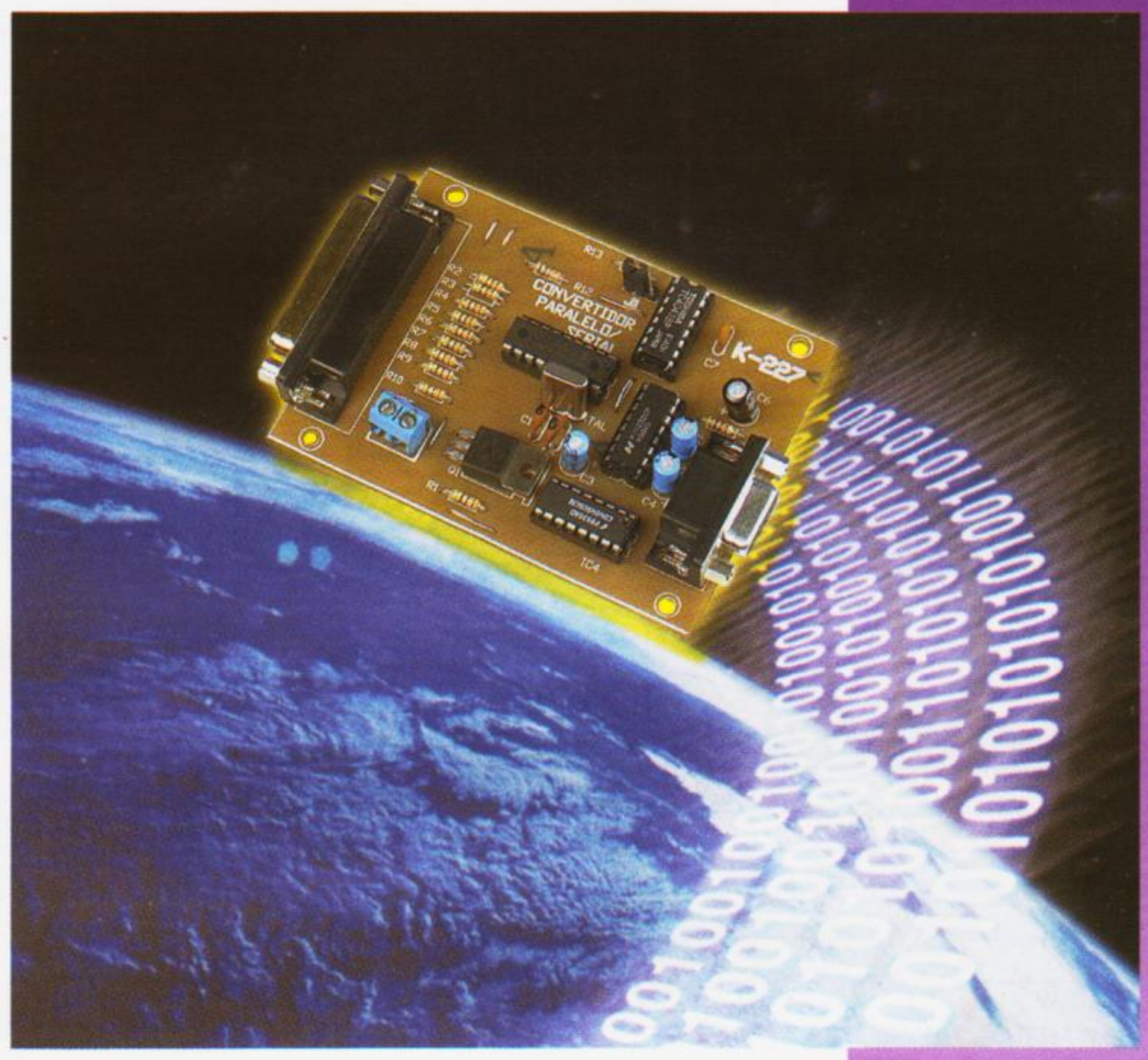
Paso 5
Instale y suelde el relé de 12V con capacidad de 10A, el cristal de cuarzo, y el condensador electrolítico C1.
Figura 22.15



Paso 6
Aspecto final de la alarma ensamblada, luego de conectar el LED indicador, e insertar el microcontrolador en su base.
Figura 22.16

Paso 7. Pruebas finales
Si la llave de encendido está cerrada el sistema no ejecutará ninguna función. Si se hace la activación de la alarma, de acuerdo a lo expuesto en párrafos anteriores, el LED se apagará y se vigilará constantemente el estado del sensor. Si en ese momento la motocicleta se mueve, el LED se encenderá y el relé se activará y desactivará continuamente, hasta que se cierre de nuevo el encendido.

PROYECTO 23



CONVERTIDOR PARALELO-SERIAL

Existen casos en donde no está libre un puerto serial para trabajar en algunos desarrollos propios. Este circuito permite trabajar el puerto paralelo como un puerto serial adicional.

Características

- Conversión de datos en paralelo, a formato serial
- Configuración de la velocidad serial por medio de un *jumper*
- Velocidades de 9.600 o 19.200 bps
- Conversión unidireccional, de paralelo a serial
- Apropiado para desarrollos donde no se disponga de un puerto serial

El puerto paralelo de las computadoras tiene como propiedad permitir la entrada y salida de datos digitales. El circuito presentado en esta ocasión, consiste en un sistema capaz de tomar la información enviada a través del puerto paralelo y convertirla en información que cumple con el protocolo serial RS-232. Este proyecto puede ser de gran utilidad en sus tareas específicas de control o instrumentación, donde no disponga de un puerto serial para trabajar, o en casos donde requiera disminuir el número de líneas de comunicación.

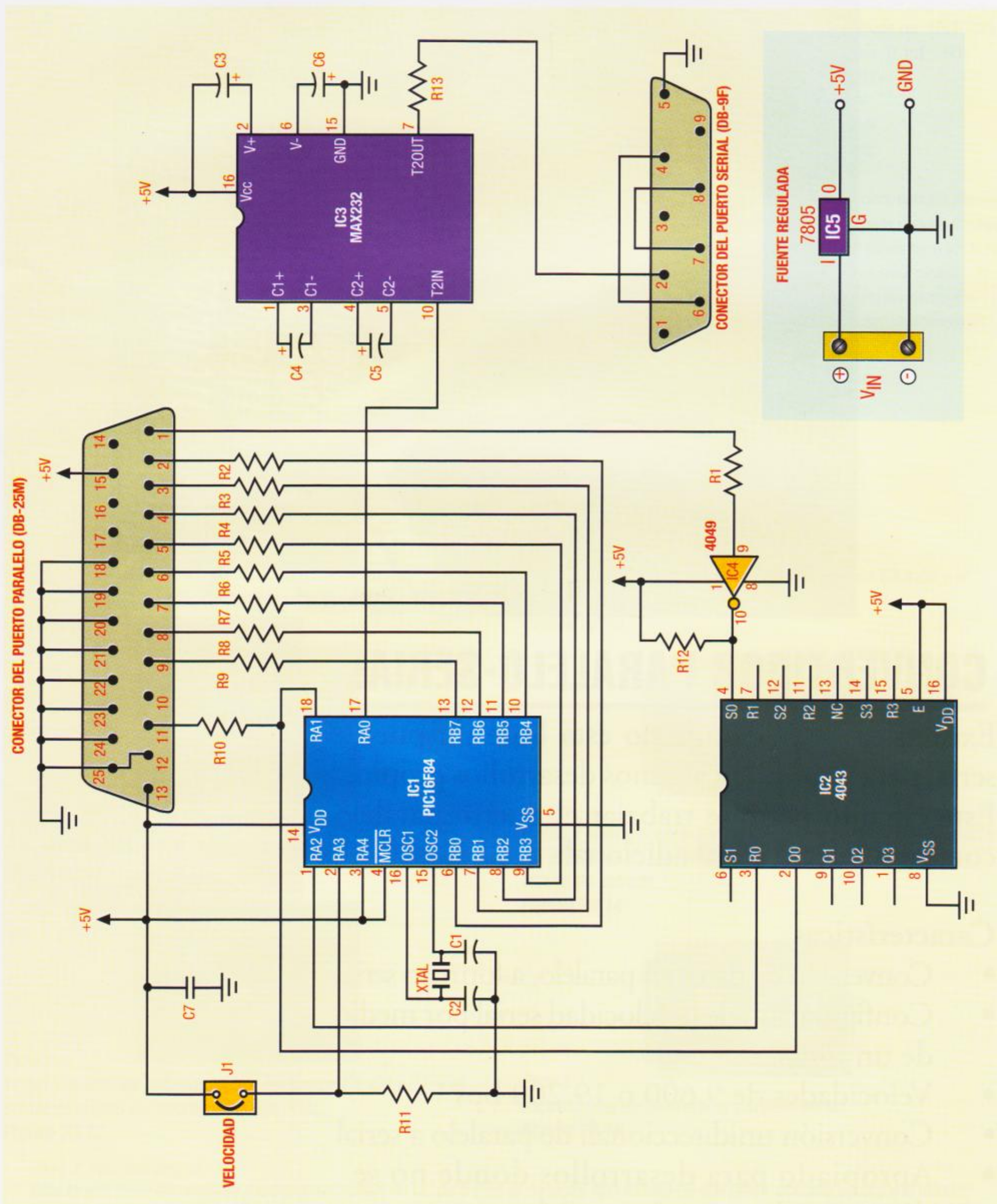


Figura 23.1 *Diagrama esquemático del circuito*

Pin D-sub	Señal	Función
1	nStrobe	Strobe
2	D0	*Bit* de datos 0
3	D1	*Bit* de datos 1
4	D2	*Bit* de datos 2
5	D3	*Bit* de datos 3
6	D4	*Bit* de datos 4
7	D5	*Bit* de datos 5
8	D6	*Bit* de datos 6
9	D7	*Bit* de datos 7
10	nAck	Reconocimiento
11	Busy	Impresora ocupada
12	PaperEnd	Sin papel
13	Select	Impresora seleccionada
14	nAutoLF	Genera automáticamente saltos de página
15	nError (nFault)	Error
16	nInit	Inicializar impresora
17	nSelectIn	Seleccionar impresora
18	Gnd	Tierra para D0
19	Gnd	Tierra para D1, D2
20	Gnd	Tierra para D3, D4
21	Gnd	Tierra para D5, D6
22	Gnd	Tierra para D7
23	Gnd	Tierra para nSelectIn
24	Gnd	Tierra para Busy
25	Gnd	Tierra para nInit

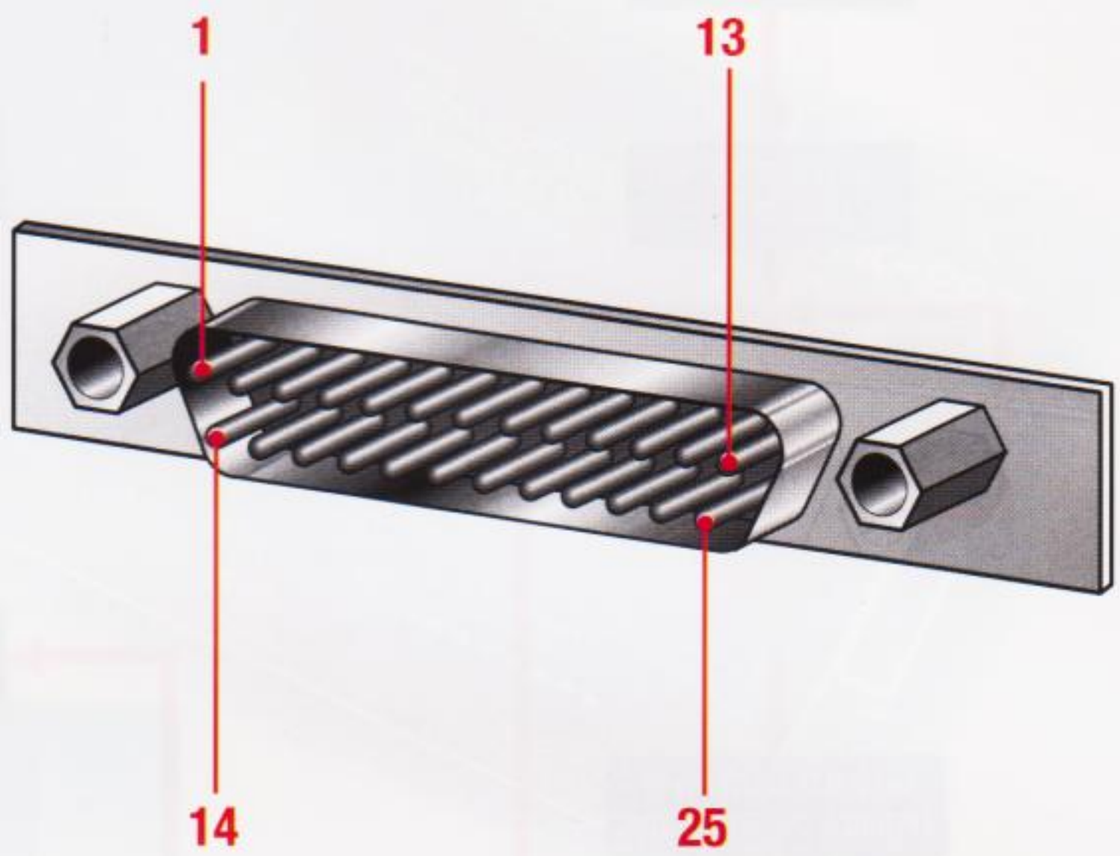


Figura 23.2 *Diagrama de pines y tabla de funciones del puerto paralelo*

La conversión es **unidireccional**, es decir, de paralelo a serial. Se trata de un circuito que lee la información enviada desde el puerto paralelo y luego la envía de manera serial.

Funcionamiento del circuito

En la **figura 23.1** aparece el diagrama esquemático del circuito diseñado para el envío de señales a través del puerto paralelo y su conversión a protocolo serial, llegando finalmente a un conector DB9.

El corazón del proyecto es un microcontolador PIC, el cual se adhiere al protocolo de comunicación paralela. De esta manera, cuando los pines de datos del puerto paralelo tienen listo un *byte*, se envía una señal, llamada *strobe*, de reconocimiento o de aprobación desde la computadora, la cual debe tener muy poca duración (unos cuantos microsegundos).

El microcontolador no debe operar hasta no tener seguridad en la comunicación; por esta razón se introdujo un *flip-flop*. El pulso de *strobe* o de reconocimiento viene del *flip-flop* y da el pulso para aprobar la comunicación.

La **figura 23.2** muestra las funciones de cada uno de los 25 pines del puerto paralelo o puerto de impresora, y la información de sus señales y sus correspondientes registros; note la ubicación del pin *busy* que es el que se utiliza para permitir el flujo de los datos al sistema.

Recuerde que el puerto paralelo estándar utiliza tres registros de ocho *bits* en una PC convencional. La computadora accede a las señales del puerto paralelo al leer y escribir en estos registros, denominados los registros de datos, estados y control.

Durante el tiempo requerido para que el sistema tome el dato del puerto paralelo y lo transforme en formato serial, no se debe enviar ningún nuevo dato, en ese sentido se debe utilizar el pin de *busy*. Conectando la salida del *flip-flop* directamente a este pin, se asegura que ningún otro *byte* se enviará mientras se esté procesando el dato actual.

Una vez terminada toda la transmisión, el microcontrolador origina un pulso que reiniciará al *flip-flop*. Así, el pin *busy* quedará igualmente reiniciado y todo el procedimiento se repetirá por cada *byte* que llegue desde el puerto paralelo.

Figura 23.3 *Cuerpo principal del programa*

La salida del microcontrolador va hacia un circuito integrado MAX232 (o compatible), el cual convierte la señal de acuerdo a las especificaciones dadas en la norma internacional para comunicaciones RS-232.

Descripción del programa

El programa básicamente lee el dato del puerto B del microcontrolador y, según el *jumper* de selección de velocidad de transmisión, salta a la rutina indicada de envío, de 19.200 bps ó de 9.600 bps. La selección de la velocidad se hace a través del *jumper* J1. Si J1 se encuentra

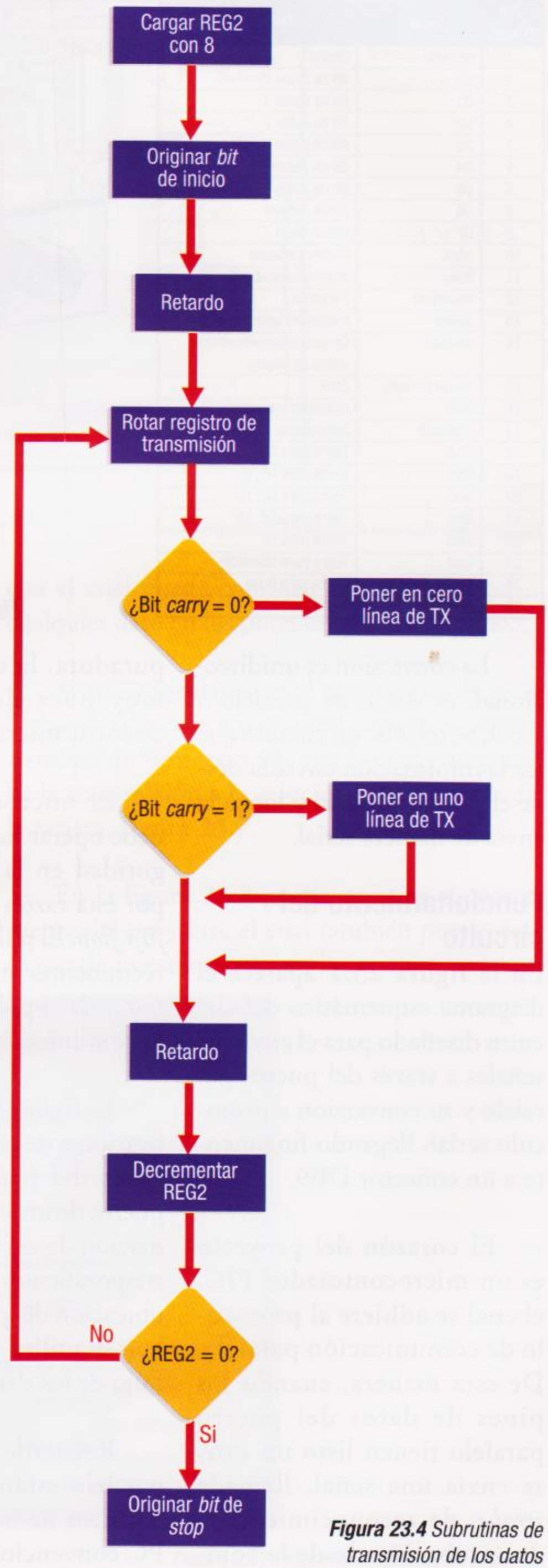


Figura 23.4 *Subrutinas de transmisión de los datos*

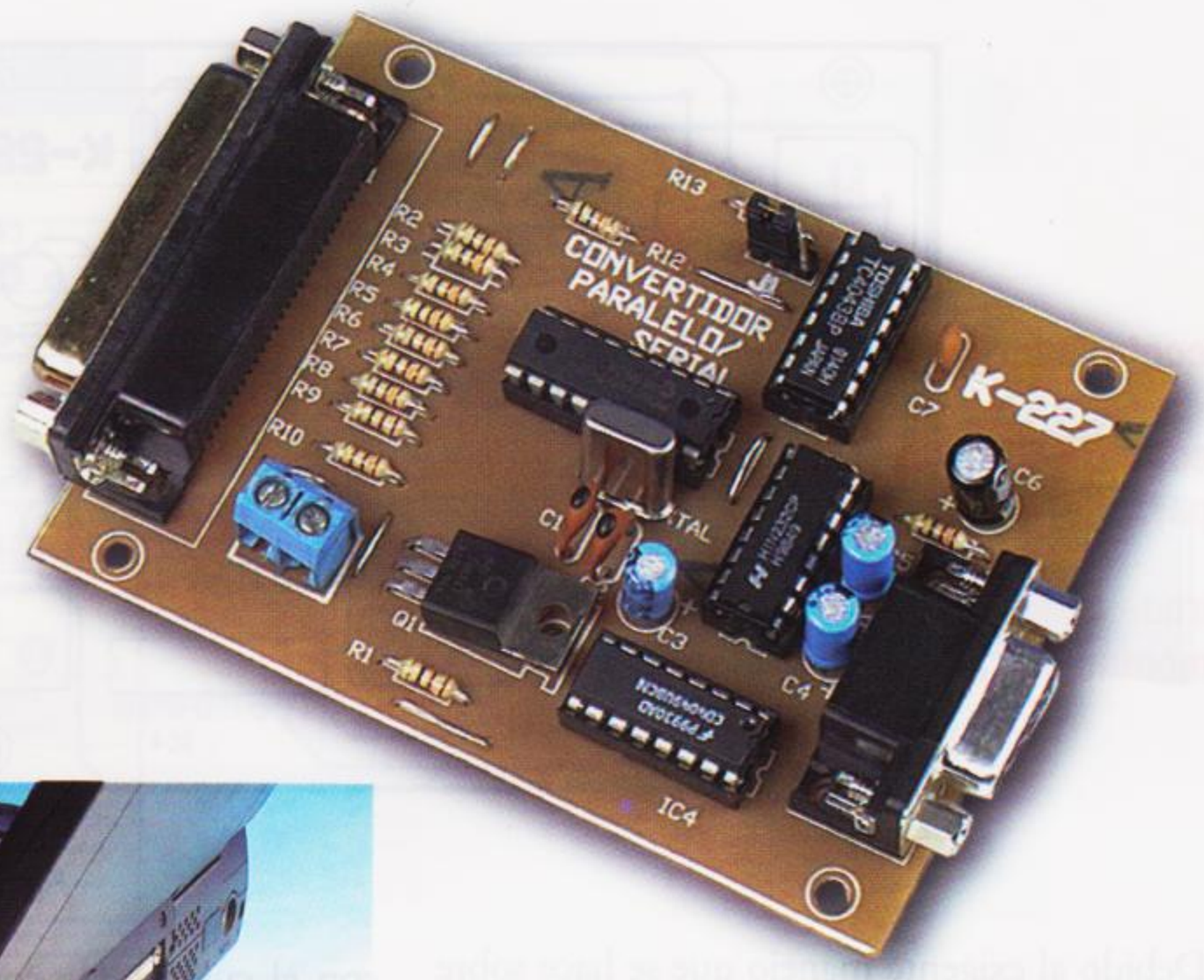


Figura 23.5 *Aspecto final de la tarjeta ensamblada y la conexión con el puerto paralelo de la computadora*

abierto se establece una velocidad de 19.200 bps y si está cerrado se trabaja a 9.600 bps. Ambas opciones sin paridad, con 8 *bits* de datos y sin ningún control de flujo.

En la **figura 23.3** puede apreciar el diagrama de flujo del cuerpo principal del proyecto; en él puede ver que la tarjeta mantiene la línea de transmisión en estado IDLE hasta que se detecte la llegada de un dato válido desde el puerto paralelo. Ese dato se almacena en un registro especializado denominado registro de transmisión TXREG y se procede a leer el *jumper* que configura la velocidad a la cual va a transmitir.

De acuerdo a la velocidad seleccionada, el programa llama una de las rutinas de transmisión, hay una para transmitir a 9.600 bps y otra para transmitir a 19.200 bps.

En la **figura 23.4** puede ver el diagrama de flujo genérico para cada una de las subrutinas de transmisión.

En él puede observar que inicialmente se envía el *bit* de inicio y sobre el registro de transmisión se efectúan ocho rotaciones en cada una de las cuales se determina el estado del *bit carry* del registro de estado, y según dicho estado, se envía el nivel al pin de transmisión. Finalmente se envía el *bit* de parada o STOP.

El aspecto final del proyecto lo puede apreciar en la **figura 23.5**. En realidad puede ver que el resultado final es una tarjeta que ocupa poco espacio y se puede instalar en el puerto de la computadora, en su parte posterior.

Para el montaje de este circuito, igual como lo hemos indicado en anteriores proyectos, debe instalar primero los puentes de alambre y los componentes de menor tamaño, dejando por último los conectores de cada uno de los puertos. Utilice como guía de montaje el diagrama de la **figura 23.6**.

Verifique que las soldaduras hayan quedado firmes y que no hagan contacto con los puntos contiguos.

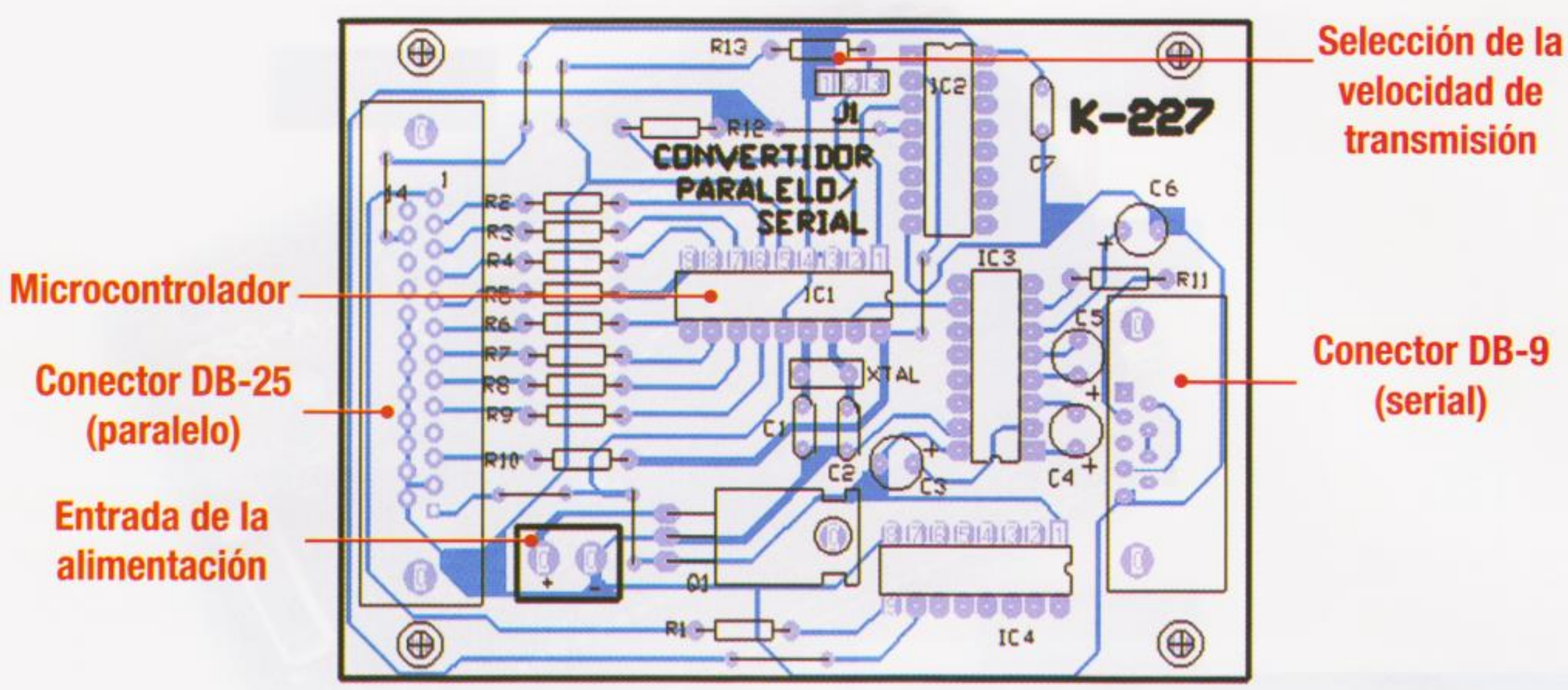


Figura 23.6 Guía de montaje del proyecto

Debido al exigente manejo que se hace sobre el reloj, es mejor que se asegure de limpiar la placa del circuito impreso, para que los residuos de resina y de soldadura no queden ejerciendo falsos contactos lo que alteraría el funcionamiento general del circuito. Este proyecto con todos sus componentes lo puede adquirir en cualquiera de nuestros distribuidores bajo la referencia K-227 de CEKIT.

Pruebas del sistema

Para comprobar el correcto funcionamiento de este proyecto, se acudió a un programa en Visual Basic con el cual puede controlar el estado y escribir cualquier dato en los pines del puerto paralelo.

Al tener los datos en el puerto, puede recibirlos serialmente a través del programa *hyperterminal* de Windows 95, con lo cual comprueba la transmisión a la velocidad configurada en la tarjeta, **figura 23.7**.

En la **figura 23.8** puede apreciar el código fuente del proyecto, el cual también puede descargar de nuestra página en la internet.

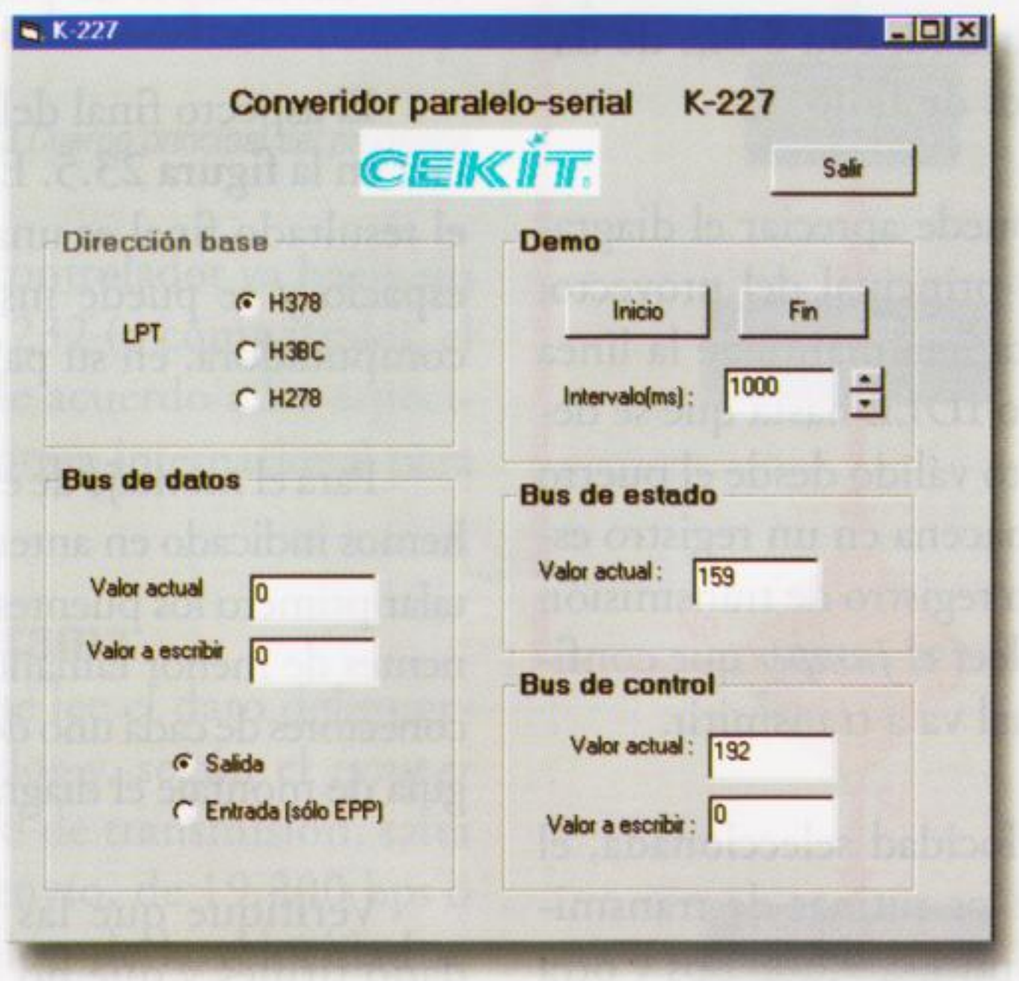


Figura 23.7 Programa de prueba de la tarjeta

```
RTCC   EQU      01H                          ; CURSO PRÁCTICO SOBRE MICROCONTROLADORES****
STAT   EQU      03H                          ; CEKIT- PEREIRA - COLOMBIA
PORTA  EQU      05H                          ; CONVERTIDOR PARALELO / SERIAL
PORTB  EQU      06H                          ; AUTOR JUAN ANDRES CASTAÑO WELGOS
INTCON EQU      0BH
TR_A   EQU      85H
TR_B   EQU      86H
AREG01 EQU      0CH
AREG02 EQU      0DH
TXREG  EQU      0EH                          ; REGISTRO DE TRANSMISIÓN
#define TX  PORTA,0                          ; PIN DE TRANSMISIÓN
       ORG              0
       goto             START                ;VECTOR DE RESET
       nop
       goto             INTERP               ;SERVICIO DE INTERRUPCIÓN
; ------------------------- RUTINAS DE TRANSMISION SERIAL DE DATOS A 19200 BPS ------------------------------
DEL192 movlw           .10                   ; SE REPITE 10 VECES
       movwf            AREG01               ; W SE USA COMO CONTADOR
LOOP19 clrwdt                                ; BORRAR PERRO
       decfsz           AREG01               ; DECREMENTAR REG1
       goto             LOOP19               ;
       clrwdt                                ; RETARDO
       clrwdt
       return
TXRS19 movlw           .8                    ; REPETIR TODO PARA LA 8 BITS
       movwf           AREG02                 ; AREG02 MANTIENE LA CUENTA DE BITS
       bcf             TX                     ; BIT DE INICIO
       call            DEL192                 ; RETARDO
LOOPT1 rrf             TXREG                  ; SIGUIENTE BIT
       btfsc           STAT,0                 ; SI EL BIT DE TX ES 1
       bsf             TX                     ; ACTIVE  EL BIT DE TX = 1
       btfss           STAT,0                 ; SI EL BIT DE TX ES 0
       bcf             TX                     ; PONGO EL PIN DE Tx EN 0
       call            DEL192                 ; RETARDO
       decfsz          AREG02                 ; DECREMENTO AREG02 Y PARO SI NO
       goto            LOOPT1                 ; QUEDA NINGÚN BIT
       bsf             TX                     ; BIT DE STOP
       call            DEL192                 ;
       nop                                    ; RETARDO
       return                                 ;
;-------------------------RUTINAS PARA TRANSMISION DE DATOS A 9600 BPS-----------------------------------
DEL96  movlw           .23                    ; REPETIR 10 VECES
       movwf           AREG01                 ; W COMO CONTADOR
LOOP96 clrwdt                                 ; BORRAR EL PERRO
       decfsz          AREG01                 ; DECREMENTAR Y REPETIR HASTA 0
       goto            LOOP96                 ;
       clrwdt                                 ; RETARDO
       clrwdt                                 ;
       return                                 ; 95 9+Xx4     80
TXRS96 movlw           .8                     ; REPETIR TODO PARA LOS 8 BITS
       movwf           AREG02                 ; AREG02 MANTIENE LA CUENTA DE BITS
       bcf             TX                     ; BIT DE INICIO
       call            DEL96                  ; RETARDO
LOOPT9 rrf             TXREG                  ; SIGUIENTE BIT
       btfsc           STAT,0                 ; SI EL BIT DE TX ES 1
       bsf             TX                     ; ACTIVE  EL BIT DE TX = 1
       btfss           STAT,0                 ; SI EL BIT DE TX ES 0
       bcf             TX                     ; PONGO EL PIN DE Tx EN 0
       call            DEL96                  ; RETARDO
       decfsz          AREG02                 ; DECREMENTO AREG02 Y PARO SI NO
       goto            LOOPT9                 ; QUEDA NINGUN BIT
       bsf             TX                     ; BIT DE STOP
       call            DEL96                  ; RETARDO
       nop
       return                                                  ***********************************
START  bsf             STAT,5                 ; SELECCIONO BANCO 1 DE RAM ***INICIO DEL PROGRAMA**********
       movlw           H'FF'                  ; RB0...RB7 COMO SALIDAS   ***********************************
       movwf           TR_B
       movlw           H'0A'                  ; RA0,RA2 SALIDAS   , RA1, RA3 ENTRADAS
       movwf           TR_A
       clrwdt                                 ; BORRO PERRO GUARDIÁN
       clrf            INTCON                 ; DESHABILITO INTERRUPCIONES
       bcf             STAT,5                 ; SELECIONO BANCO 0 DE RAM
       movlw           .128                   ; CARGO EL TMR0
       movwf           RTCC
MAIN   bsf             TX                     ; PONGO LA LÍNEA SERIAL EN ESTADO IDLE
       bsf             PORTA, 2               ; BORRO LA SALIDA DEL FLIP-FLOP
       bcf             PORTA, 2
SIGO   clrwdt                                 ; BORRO PERRO GUARDIÁN
       btfss           PORTA, 1               ; ESPERO A QUE SE RECIBA UN BYTE DEL PUERTO
       goto            SIGO                   ; PARALELO.
       movf            PORTB, 0               ; GUARDO EL BYTE DE DATOS EN W
       movwf           TXREG                  ; LO ALMACENO EN EL BYTE DE TX
       btfss           PORTA, 3               ; LEO LA CONFIGURACIÓN DE LA VELOCIDAD
       call            TXRS19                 ; Y LLAMO LA RUTINA DE TRANSMISION
       btfsc           PORTA, 3               ; LEO LA CONFIGURACIÓN DE LA VELOCIDAD
       call            TXRS96                 ; Y LLAMO LA RUTINA DE TRANSMISION
       bsf             PORTA, 2               ; BORRO FLIP FLOP
       bcf             PORTA, 2
       goto            SIGO                   ; REPITO TODO
INTERP retfie                                 ; VECTOR DE INTERRUPCIÓN
       END
```

Figura 23.8 *Código fuente del proyecto*

Ensamblaje del proyecto

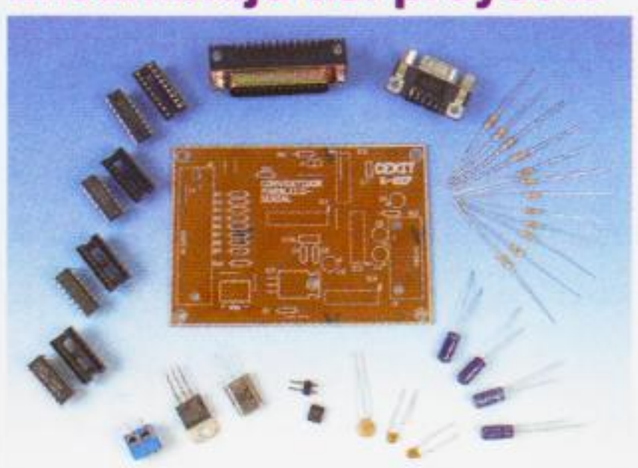
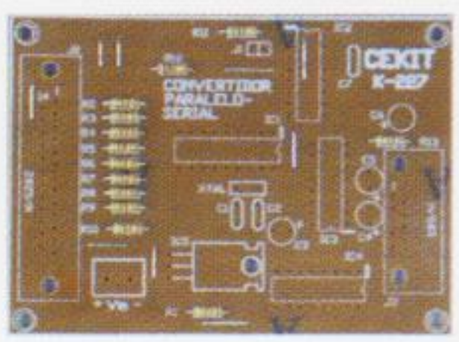


Figura 23.9 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el convertidor paralelo-serial CEKIT K-227.*

Algunos equipos electrónicos, como impresoras, escáners y computadoras personales, sólo disponen de una clase de puerto serial o paralelo. Para facilitar la comunicación y aumentar la flexibilidad de estos equipos se ha diseñado el convertidor paralelo-serial, que convierte el puerto paralelo de una computadora en un puerto serial. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

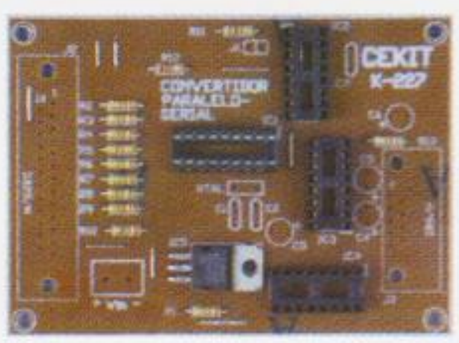
Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC 16F84, programado
1	IC2	Circuito integrado CD4043
1	IC3	Driver de comunicación MAX232
1	IC4	Circuito integrado CD4049
1	IC5	Regulador fijo de +5V, LM7805
2	C1 y C2	Condensadores cerámicos de 22pF - 50V
4	C3,C4,C5 y C6	Condensadores electrolíticos de 10uF - 25V
1	C7	Condensador cerámico de 0,1uF - 50V
11	R1 a R10 y R13	Resistencias de 220 - 1/4W
1	R11	Resistencia de 1k - 1/4W
1	R12	Resistencia de 10k - 1/4W
1	XTAL	Cristal de cuarzo de 4MHz
1	K-227	Circuito impreso CEKIT
1	CON1	Conector DB 25 Macho para circuito impreso
1	CON2	Conector DB 9 Hembra para circuito impreso
1		Conector de tornillo de 2 pines para impreso
1		Conector ribbon tipo cerca de 2 pines
1		Base de 18 pines para circuito integrado
3		Bases de 16 pines para circuito integrado
1		Jumper



Paso 1

Instale y suelde los puentes de alambre y las resistencias R1 a R13.

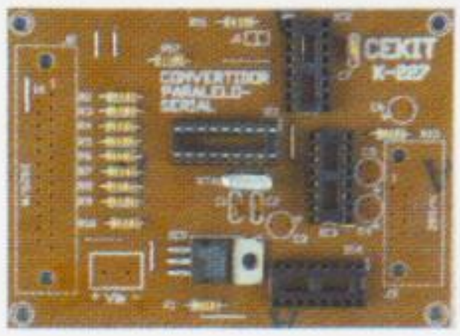
Figura 23.10



Paso 2

Luego instale y suelde las bases para el microcontrolador y los circuitos integrados, y el regulador integrado de +5V IC5.

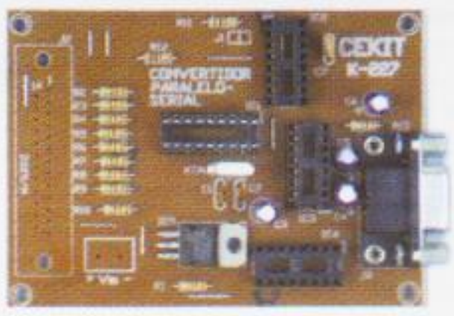
Figura 23.11



Paso 3

Instale y suelde los condensadores cerámicos C1, C2 y C7, y el cristal de cuarzo del microcontrolador XTAL.

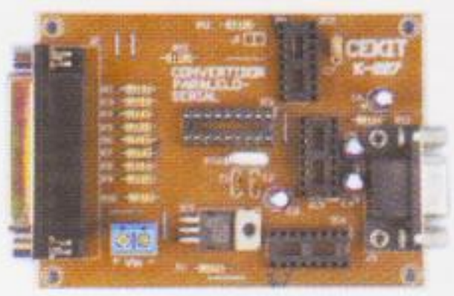
Figura 23.12



Paso 4

Instale y suelde los condensadores electrolíticos C3 a C6 y el conector serial DB-9 hembra J3.

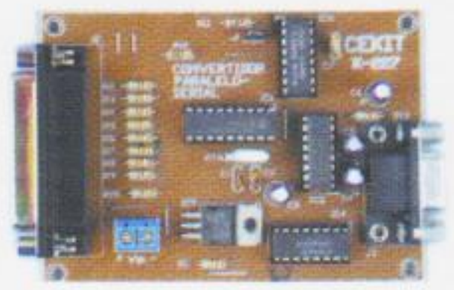
Figura 23.13



Paso 5

Instale y suelde el conector para el puerto paralelo DB-25 macho J2, el jumper J1, y el conector de tornillo de 2 pines para el voltaje de entrada.

Figura 23.14



Paso 6

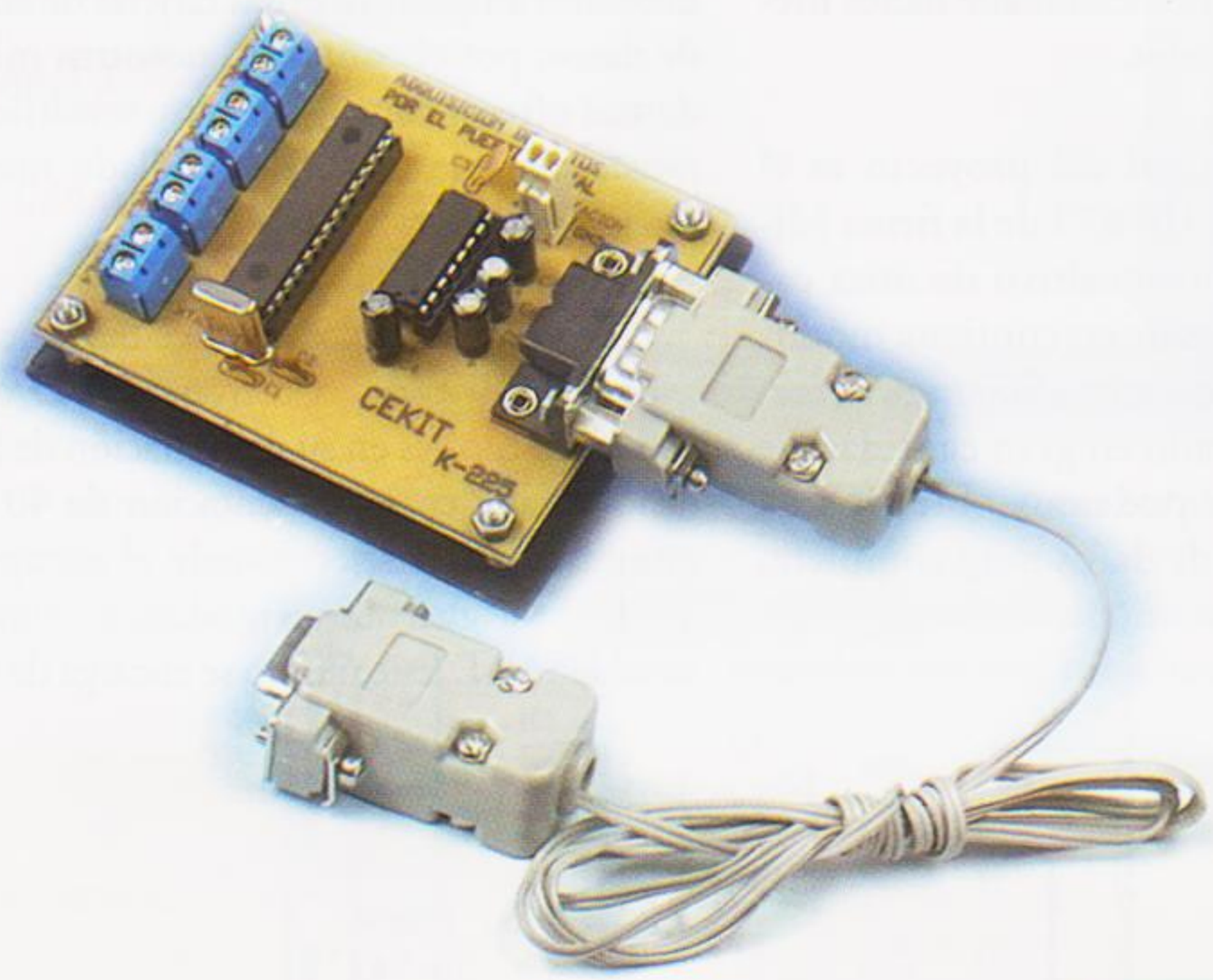
Aspecto final del convertidor paralelo-serial.

Figura 23.15

Paso 7. Pruebas finales

Esta tarjeta debe alimentarse mediante un voltaje de CC de 5 V en el conector tipo tornillo de dos pines, ubicado en la parte inferior. Conecte adecuadamente la alimentación para evitar daños en los circuitos integrados. Por medio de un programa de prueba de PC, envíe datos desde el puerto paralelo de la tarjeta y verifique la transmisión de datos en el pin TX del conector DB-9.

PROYECTO 24



ADQUISICIÓN DE DATOS POR EL PUERTO SERIAL

Ésta es una económica y versátil tarjeta de adquisición de datos por el puerto serial que utiliza un microcontrolador PIC16F873, lo que nos permite medir señales de entrada en datos de 10 *bits*. Es de gran utilidad para el estudio de la tecnología de adquisición de datos a nivel académico o a título personal. También se puede utilizar en la industria para el monitoreo y control de todo tipo de variables.

Características

- Adquisición de cuatro señales análogas previamente acondicionadas de cero a cinco voltios
- Tamaño reducido
- Conexión por el puerto serial
- Cada señal se adquiere en 10 *bits*
- Transmisión a 2.400 bps

Ahora construiremos un dispositivo que nos permitirá conocer el valor de cuatro señales análogas provenientes de cualquier sensor, las cuales deben estar previamente acondicionadas para que se encuentren en el rango de cero a cinco voltios. Esta precaución es vital para no ocasionarle daños irreversibles al microcontrolador.

El elemento principal del proyecto es el microcontrolador PIC 16F873 de la firma Microchip, el cual es un dispositivo de unas características muy interesantes: contiene módulos internos que lo hacen muy eficiente y atractivo para ser incorporado en gran cantidad de aplicaciones y con él, podemos obtener una tarjeta de adquisición de datos muy completa, que transmite los datos adquiridos a una buena velocidad, que presenta un tamaño reducido, y por último, que se puede construir con un bajo presupuesto.

Este proyecto nos demuestra que para muchas aplicaciones en electrónica, no siempre es necesario adquirir costosas tarjetas de adquisición de datos; por el contrario, nosotros mismos podemos ofrecer soluciones muy versátiles y con la satisfacción de haber desarrollado nuestra propia tecnología.

Entre los módulos internos de este circuito integrado se destacan el conversor A/D de 10 *bits*, con cinco canales en su presentación de 28 pines y ocho canales en su presentación de 40 pines, en este caso hemos seleccionado el encapsulado de 28 pines; al igual que un módulo de comunicación serial USART. Este último se encarga de transmitir

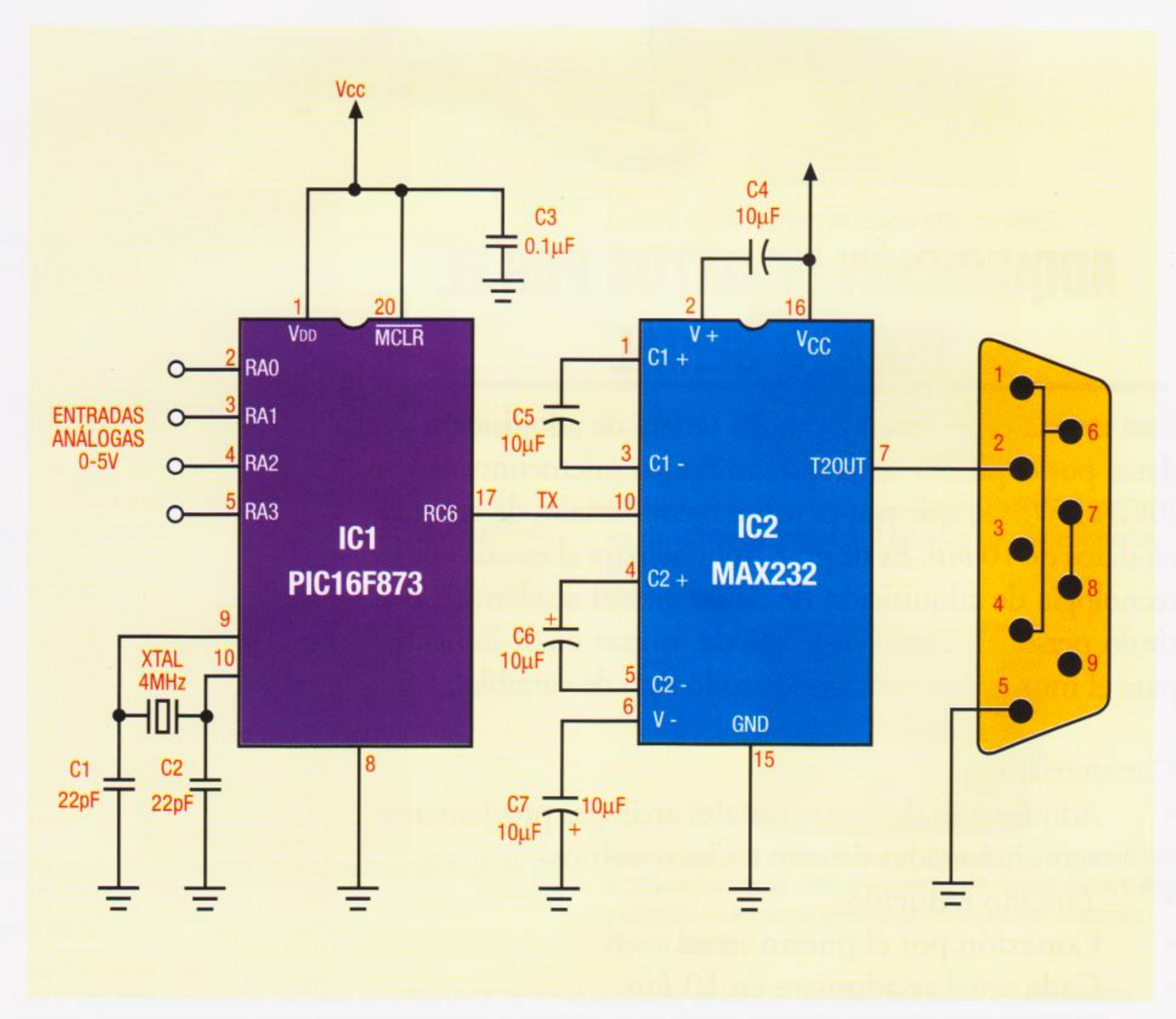


Figura 24.1 Diagrama esquemático del proyecto

el valor resultante de la conversión hacia la computadora, realizando todas las actividades necesarias para la transmisión, por lo cual únicamente nos debemos preocupar de configurar adecuadamente ciertos registros.

Ambos módulos, el conversor y el USART, se han utilizado en este proyecto y por esa razón los entraremos a analizar brevemente.

Módulo conversor análogo-digital

Recordemos el principio básico bajo el cual trabaja un conversor A/D: dependiendo de la magnitud de la señal análoga, se carga un condensador hasta cierto nivel; según dicho nivel, el condensador tarda determinado tiempo en descargarse, el cual puede medirse para calcular la magnitud de la señal análoga de entrada.

La conversión de esa entrada análoga entrega un número digital de 10 *bits*, razón por la cual, cuando la conversión A/D se ha completado, el resultado se carga en dos registros denominados ADRESH y ADRESL donde se almacenan la parte alta y baja del resultado, respectivamente.

Para utilizar el módulo conversor A/D interno, se deben configurar los pines de entrada, el voltaje de referencia y el reloj de conversión.

Para una buena conversión A/D, el reloj de la conversión

```
; Adquisición de datos a 10 bits y Tx serial vía RS-232
 ; ******** DEFINICION DE REGISTROS Y BITS UTILIZADOS
status  equ     03              ; registro de estado
ptoa    equ     05              ; direcciones de los puertos
ptob    equ     06              ; del microcontrolador
ptoc    equ     07
pclath  equ     0ah             ;
intcon  equ     0bh             ; registro de control de las
pir1    equ     0ch             ; interrupciones
rcsta   equ     18h             ; estado de la recepción
txreg   equ     19h             ; registro de transmisión
rcreg   equ     1ah             ; registro de recepción
adresh  equ     1eh
adcon0  equ     1fh             ; control de lo canales a/d
trisa   equ     85h             ; configuración de los puertos
trisb   equ     86h
trisc   equ     87h
txsta   equ     98h             ; estado de la transmisión
spbrg   equ     99h             ; velocidad de transmisón
adresl  equ     9eh
adcon1  equ     9fh             ; control de los canales a/d
rp0     equ     5               ; bits de configuración de la página
rp1     equ     6
irp     equ     7
z       equ     2               ; bit cero
c       equ     0               ; bit carry
txif    equ     4
go      equ     2
loops   equ     20h             ; registros para ajustar retardos
loops2  equ     21h
nibble1 equ     22h
nibble2 equ     23h
nibble3 equ     24h
buffer  equ     25h
; ***** EMPIEZA EL PROGRAMA
        org     00
        goto    inicio
retardo movwf       loops       ;subrutina de retardo en milisegundos
 top2   movlw       d'110'      ;el número de milisegundos llega
        movwf       loops2      ;cargado en el registro loops
 top    nop
        nop
        nop
        nop
        nop
        nop
        decfsz      loops2      ;pregunta si terminó 1 ms
        goto        top
        decfsz      loops       ;pregunta si termina el retardo
        goto        top2
        return
;Rutina de conversión A/D a 10 bits devuelve 3 datos en hexadecimal guardados
;en los registros nibble1, nibble2 y nibble3
 conver bsf         status,rp0
        movlw       b'10000010' ;selecciono Vref internos (+5 y GND)
                                ;y 5 canales A/D
        movwf       adcon1      ;datos en ADRESL y 2 bits en ADRESH
        bcf         status,rp0
        nop
        nop                     ;estas instrucciones no sirven para
        nop                     ;darle al micro el tiempo de adquisi.
        nop                     ;requerido. En este caso es de 7 µs aprox.
        nop
        nop
        bsf         adcon0,go   ;inicia conversión
        nop
        nop
consu   btfsc       adcon0,go   ;espera que termine de convertir el dato
        goto        consu
        movf        adresh,w    ;debe pasar los tres dígitos hexadec.
        movwf       nibble1     ;a los registros individuales para
        movlw       30h         ;ser transmitidos
        addwf       nibble1     ;Al registro nibble 1 le suma 30h para
        bsf         status,rp0  ;convertirlo en ASCII antes de TX
        movf        adresl,w
        bcf         status,rp0
        movwf       buffer
        andlw       0fh         ;El registro nibble3 que contiene el
        movwf       nibble3     ;dígito menos significativo
        movlw       0ah         ;debe probarse para ver si es un
```

```
        subwf   nibble3,0           ;número de 0 a 9 o si es letra A hasta F
        movlw   37h                 ;para hacer la conversión a código ASCII
        btfss   status,c
        movlw   30h
        addwf   nibble3
        swapf   buffer,0
        andlw   0fh
        movwf   nibble2             ;El dígito del centro nibble2 tiene el mismo
        movlw   0ah                 ;procedimiento que nibble3
        subwf   nibble2,0
        movlw   37h
        btfss   status,c
        movlw   30h
        addwf   nibble2
        return
enviar  movf    nibble1,0
esper   btfss   pir1,txif           ;Esta rutina envía serialmente los datos
        goto    esper               ;obtenidos en cada conversión
        movwf   txreg               ;transmite el dígito hexadecimal de mayor peso
        movf    nibble2,0           ;transmite el segundo dígito hexadecimal
espera  btfss   pir1,txif           ;pero debe esperar que termine de TX el anterior
        goto    espera
        movwf   txreg
        movf    nibble3,0           ;transmite el tercer dígito
esper2  btfss   pir1,txif           ;pero antes revisa que haya enviado el segundo
        goto    esper2
        movwf   txreg
        movlw   d'44'               ;el carácter «coma» sirve para separar el dato
esper3  btfss   pir1,txif           ;de los diferentes canales
        goto    esper3
        movwf   txreg
        return
;************  PROGRAMA PRINCIPAL  ***************
inicio  bsf     status,rp0
        movlw   0ffh
        movwf   trisa               ;puerto A entradas
        movwf   trisb               ;puerto B entradas
        movlw   b'10111111'
        movwf   trisc               ;puerto C entradas y pin TX salida
        movlw   b'00100110'         ;programa USART no sincrónico para TX
        movwf   txsta               ;alta rata de baudios
        movlw   d'103'              ;configura temporizador a 4 MHZ
        movwf   spbrg               ;2.400 bps  (8,n,1)
        bcf     status,rp0
        bsf     pir1,txif
        movlw   b'10000000'         ;habilita puerto serial
        movwf   rcsta
        movlw   b'10000001'         ;selecciona canal 0, reloj de conv. fosc/32
        movwf   adcon0              ;convertidor encendido
        call    conver              ;llama rutina de conversión A/D
        call    enviar              ;envía datos seriales
        movlw   b'10001001'         ;selecciona canal 1, reloj de conv. fosc/32
        movwf   adcon0              ;convertidor encendido
        call    conver              ;llama rutina de conversión A/D
        call    enviar              ;envía datos seriales
        movlw   b'10010001'         ;selecciona canal 2, reloj de conv. fosc/32
        movwf   adcon0              ;convertidor encendido
        call    conver              ;llama rutina de conversión A/D
        call    enviar              ;envía datos seriales
        movlw   b'10011001'         ;selecciona canal 3, reloj de conv. fosc/32
        movwf   adcon0              ;convertidor encendido
        call    conver              ;llama rutina de conversión A/D
        call    enviar              ;envía datos seriales
        movlw   d'13'               ;La separación entre cada bloque de 4 datos
esper4  btfss   pir1,txif           ;se hace con un carriage return
        goto    esper4              ;y un carácter line feed
        movwf   txreg
        movlw   d'10'               ;sirve para separar las líneas de datos que
esper5  btfss   pir1,txif           ;se reciben en el programa terminal de Windows
        goto    esper5              ;Cada línea de datos tiene canal0, canal1, canal2 y
canal3  movwf   txreg
        nop
        nop
esper6  btfss   pir1,txif           ;esta comprobación es para garantizar que se transmita
        goto    esper6              ;el último carácter antes de seguir con otra actividad
        movlw   d'250'              ;retardo entre bloques de 4 caracteres
        call    retardo             ;este retardo equivale a 250 milisegundos
        goto    inicio              ;Vuelve a empezar el ciclo para enviar otro paquete de datos
        end
```

Figura 24.2 *Código fuente del proyecto*

debe asegurar por lo menos, un tiempo de conversión de 1,6 us para cada *bit*.

Módulo USART

Varios microcontroladores PIC de las series 16Cxxx, 16Fxxx y 17Cxxx poseen un módulo USART interno o módulo *universal transmisor-receptor sincrónico-no sincrónico*. Este módulo se encarga de realizar la comunicación no sincrónica *full-duplex* o la comunicación sincrónica *half duplex*.

Este módulo, además de arrojar excelentes resultados, permite aliviar un poco el trabajo del programador y permite más espacio de programa disponible para otras tareas distintas a la de la comunicación.

El formato de los datos manejados por el módulo es: un *bit* de inicio, ocho o nueve *bits* de datos, y un *bit* de parada (*stop*). Dentro de este formato no se encuentra el *bit* de paridad, sin embargo, el noveno *bit* puede usarse para ese fin. En esta ocasión seguiremos el diagrama mostrado en la **figura 24.1**, en el cual existe comunicación con una computadora. A pesar de que en este caso se ha utilizado un PIC16F873, las rutinas son extendibles a otros dispositivos sin necesidad de hacer mayores cambios.

Existen ocho registros especiales asociados con la USART, los cuales se deben configurar convenientemente para establecer el control, el estado y la interfaz de datos que requiere el programa. Toda la configuración de estos registros se puede seguir en el programa, el cual se muestra en la **figura 24.2**.

Analicemos brevemente algunos de estos registros. El registro SPBRG se usa para establecer la rata de transmisión (*bits* por segundo); el contenido que debe tener este registro, antes de habilitar la USART, se determina según la siguiente fórmula:

$$\textbf{SPBRG} = \frac{\text{Fosc} - 64\ (\text{Ra})}{64\ (\text{Ra})}$$

Donde **Fosc** es la frecuencia de oscilación en MHz y **Ra** es la rata de los datos en *bits* por segundo.

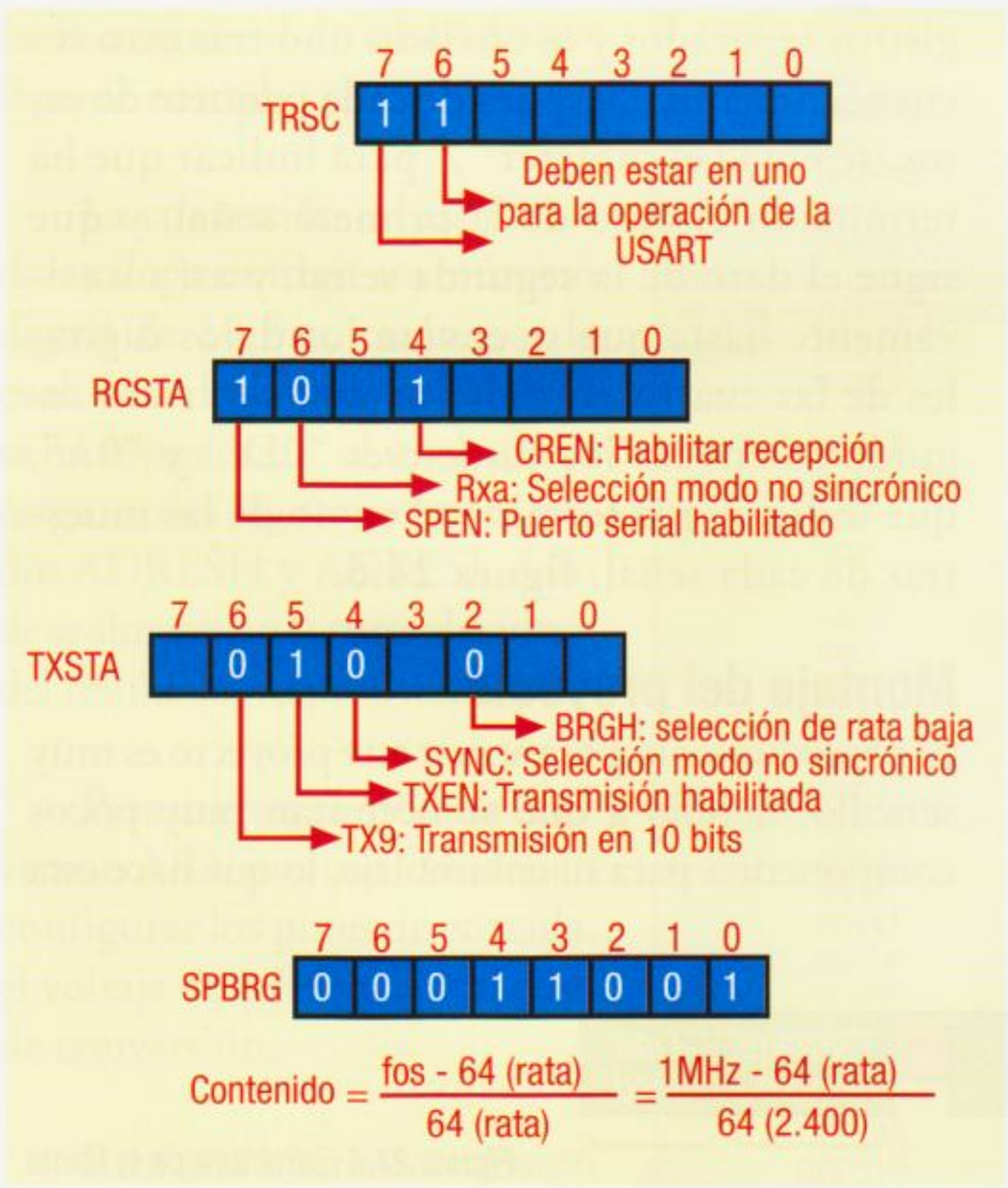


Figura 24.3 Registros asociados al control de la USART

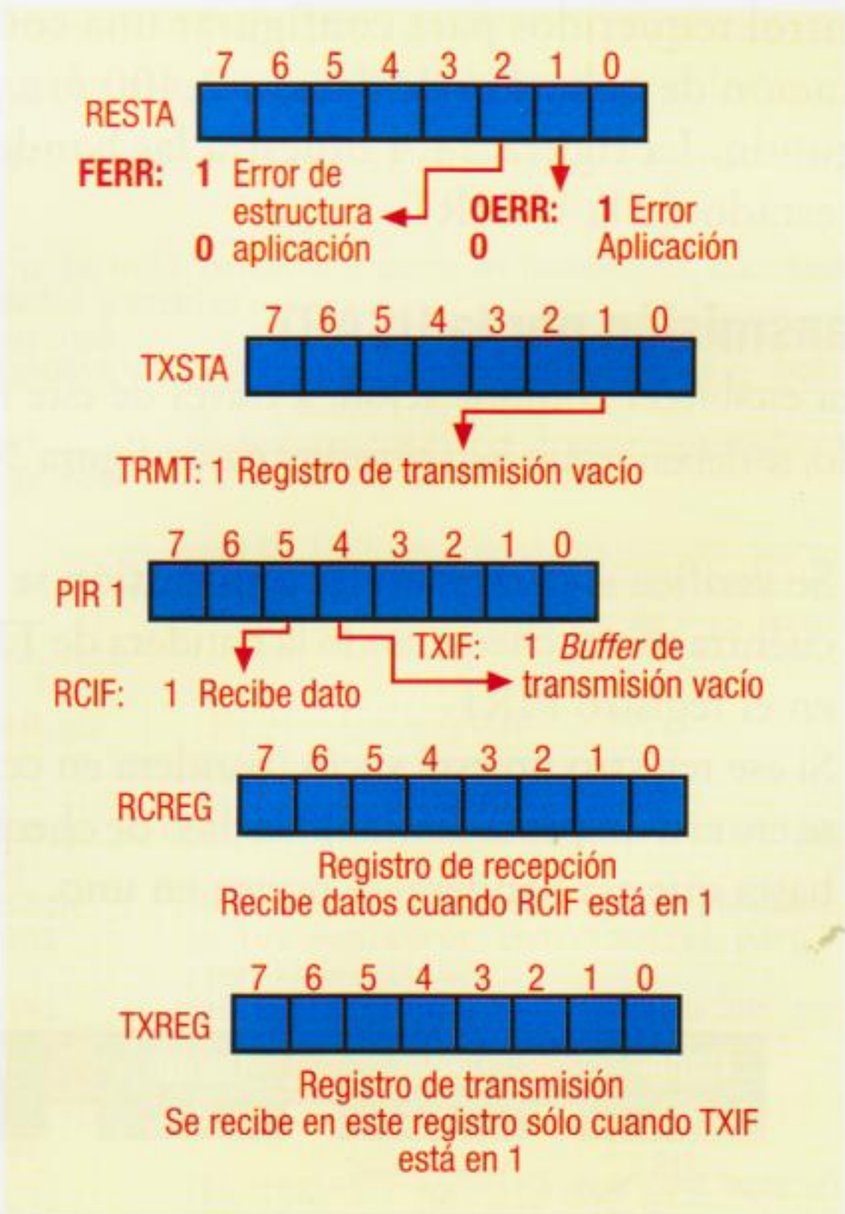


Figura 24.4 Registros asociados al estado de la USART

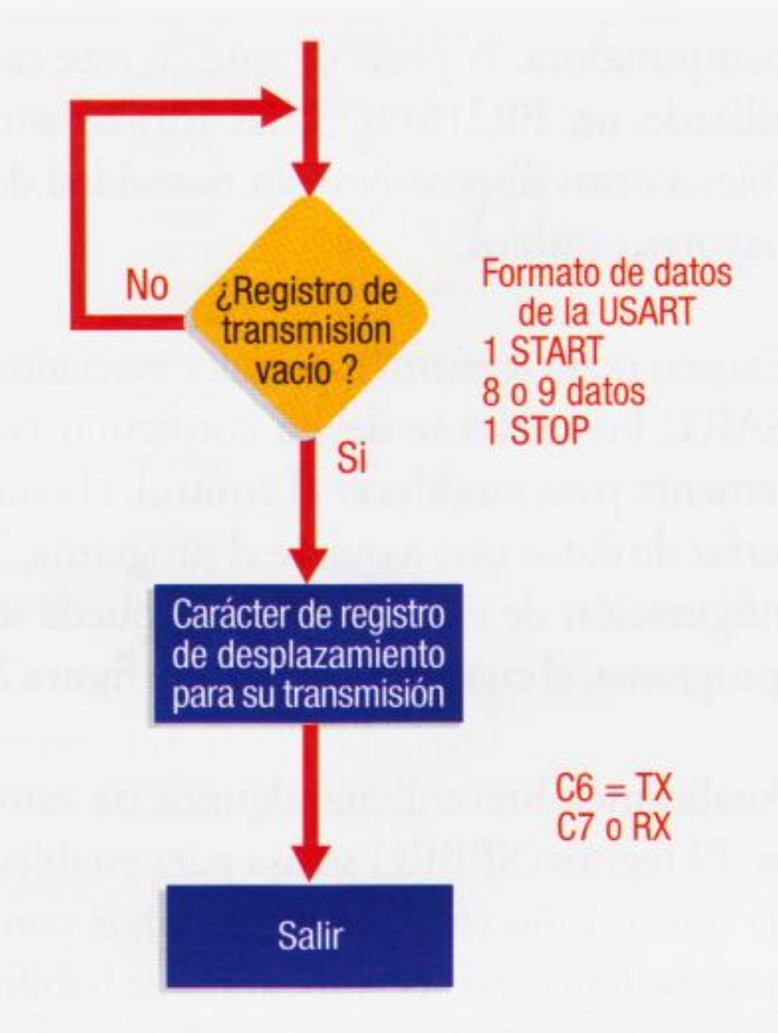


Figura 24.5 *Diagrama de flujo de la subrutina de transmisión*

Los registros RCSTA y TXSTA, establecen el estado y el control de la recepción y la transmisión, respectivamente. Los registros RCSTA, RXSTA y PIR1, proporcionan el estado de la comunicación.

En la **figura 24.3** se ven los registros de control requeridos para configurar una comunicación de ocho *bits* de datos a 2.400 *bits* por segundo. La **figura 24.4** muestra las banderas de estado de la USART.

Transmisión por la USART

Para establecer comunicación, a través de este módulo, se deben seguir los siguientes pasos, **figura 24.5**.

- Se verifica si el registro de transmisión se encuentra vacío, chequeando la bandera de TXIF en el registro PIR1.
- Si ese registro no está vacío (bandera en cero), se entra a un procedimiento cíclico de chequeo hasta que esa bandera se ponga en uno.
- Un registro vacío indica que el *byte* que va a ser transmitido ha sido movido hacia un registro de desplazamiento, que convierte el dato a formato serial y empieza el proceso de transmisión del *byte*.
- Si el registro de desplazamiento se encuentra ocupado, no tiene lugar la transmisión de nuevos contenidos, es decir, hasta que el *byte* previo haya sido transmitido completamente.

En este caso, como mencionamos en un principio, la tarjeta lee las señales provenientes de cuatro sensores ya acondicionados entre cero y cinco voltios, y esa información la envía serialmente de la siguiente forma: dentro del programa existe una rutina de conversión A/D que, como resultado final, arroja cada dato de la señal en tres caracteres hexadecimales; por ejemplo 2F5, donde el primer carácter enviado corresponde al MSB y el último al LSB.

Cada carácter de estos se almacena en registros separados y es enviado uno tras otro secuencialmente. Después de cada paquete de estos, se envía el carácter "," para indicar que ha terminado el dato de la primera señal, y que sigue el dato de la segunda señal, y así sucesivamente hasta que se envían los datos digitales de las cuatro entradas analógicas. Acto seguido se envían los caracteres "0D" y "0A", que indican que terminó el envío de las muestras de cada señal, **figura 24.6**.

Montaje del proyecto

El procedimiento para armar este proyecto es muy sencillo, debido a que se necesitan muy pocos componentes para su ensamblaje, lo que hace esta

Figura 24.6 *Estructura de la trama de transmisión*

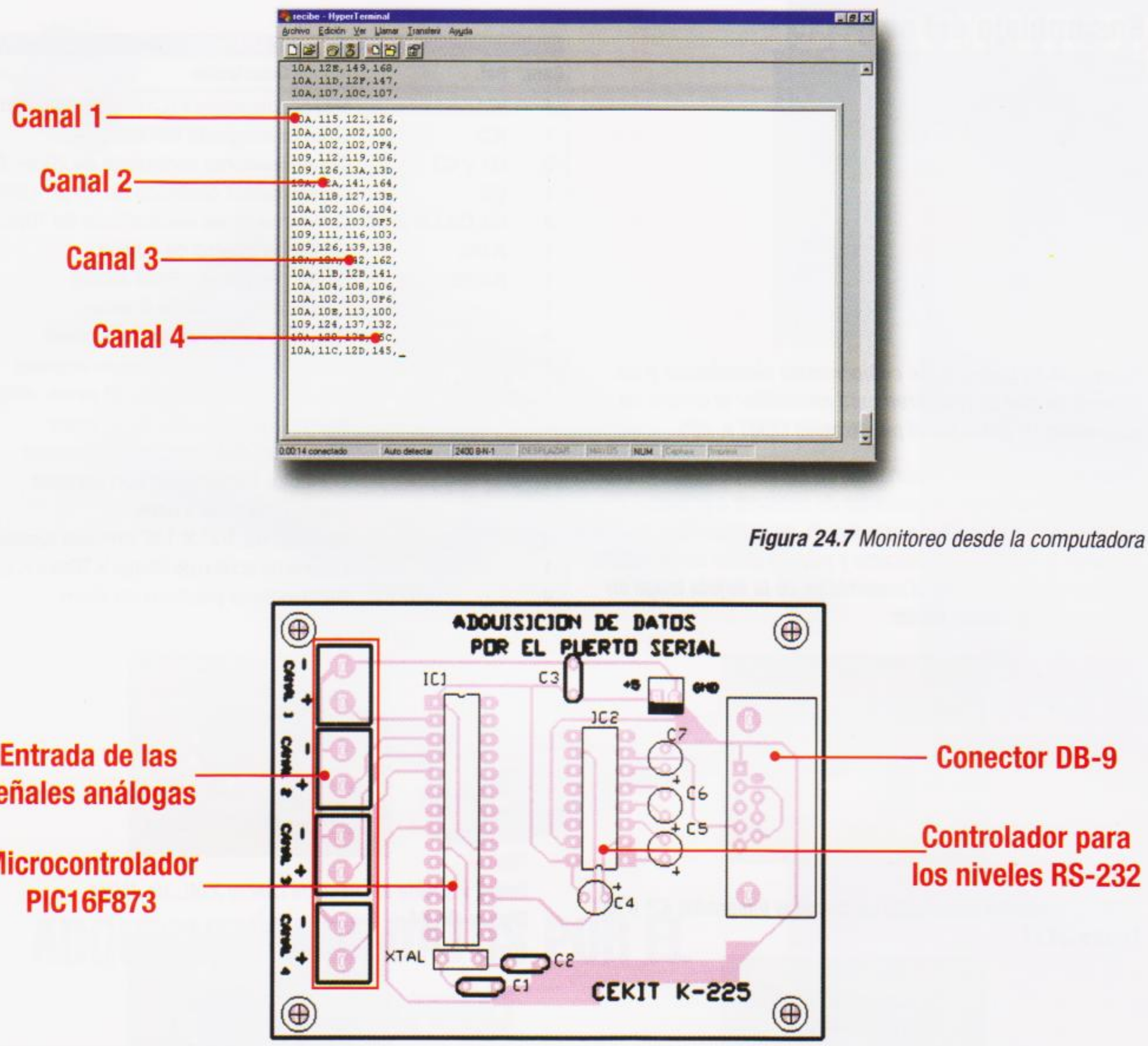


Figura 24.7 Monitoreo desde la computadora

Figura 24.8 Guía de montaje de la tarjeta

tarjeta especialmente atractiva para tenerla en nuestro laboratorio, por el poco espacio que ocupa y por la capacidad de adquirir datos en 10 *bits*. Para manejar la tarjeta se puede elaborar un programa en C, LabVIEW, o cualquier otro entorno de programación, en donde se pueda implementar la lectura teniendo en cuenta el formato con que se han enviado los datos. Sin embargo, la prueba más rápida que se puede hacer del funcionamiento de la tarjeta es a través del programa *hyperterminal* de Windows 95; éste se debe configurar a 2.400 bps, sin control de flujo, para poder leer sin ningún problema los datos que se están adquiriendo, **figura 24.7**. En la **figura 24.8** se muestra la guía de montaje y en la **figura 24.9** la apariencia final del circuito.

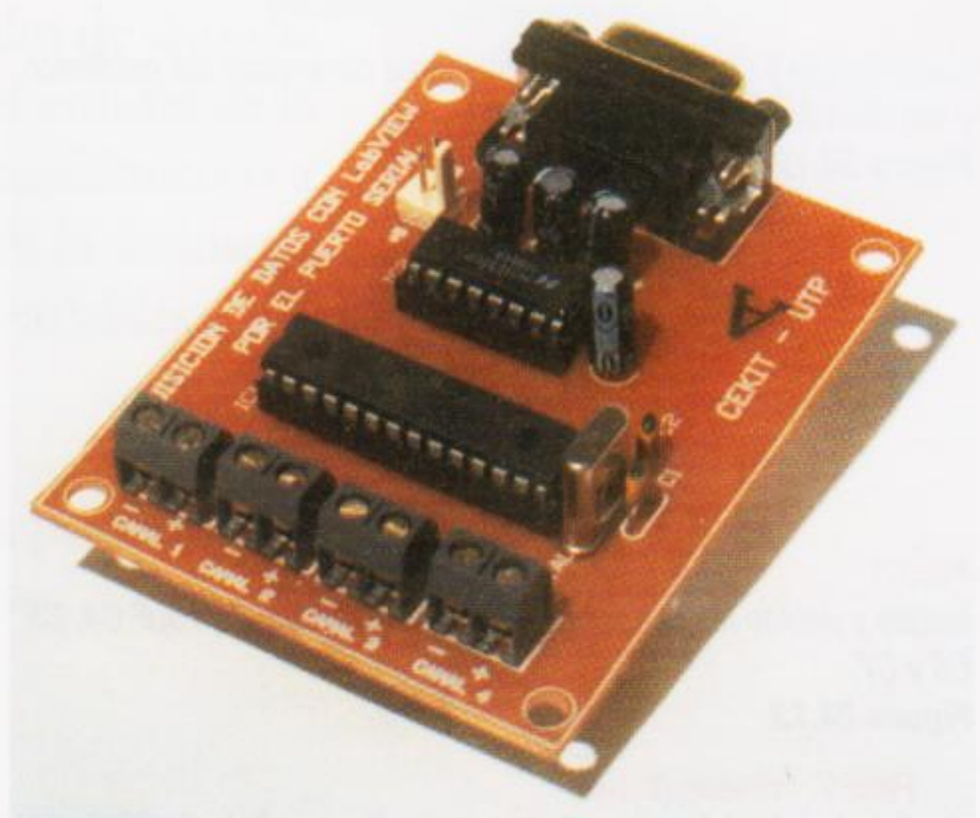


Figura 24.9 Aspecto final de la tarjeta ensamblada

Ensamblaje del proyecto

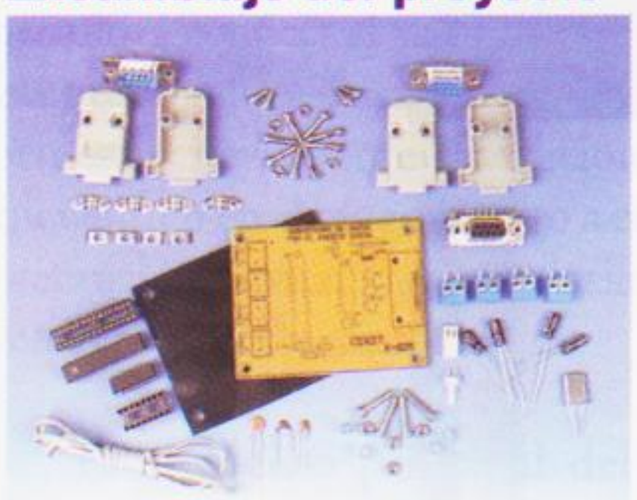
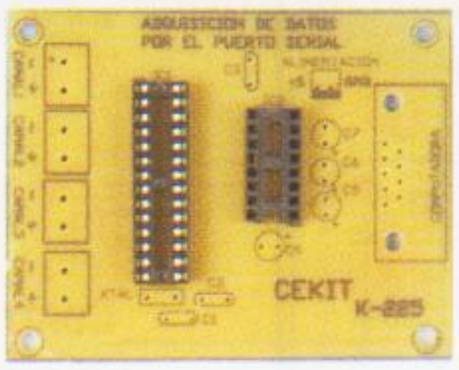


Figura 24.10 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el circuito de adquisición de datos por el puerto serial CEKIT K-225.*

Unos de los usos más importantes de las computadoras es la adquisición de datos provenientes de sensores o equipos electrónicos. Por medio de esta tarjeta, se podrá utilizar el puerto serial de la PC para comunicarse y adquirir datos desde cuatro fuentes diferentes. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

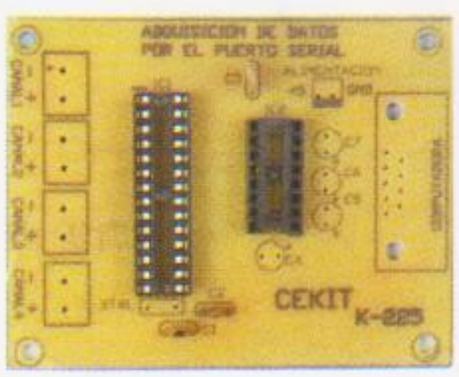
Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F873, programado
1	IC2	Circuito integrado MAX232
2	C1 y C2	Condensadores cerámicos de 20pF- 50V
1	C3	Condensador cerámico de 0,1µF - 50V
4	C4,C5,C6 y C7	Condensadores electrolíticos de 10µF/25V
1	XTAL	Cristal de cuarzo de 4 MHz
1	K-225	Circuito impreso CEKIT K-225
1		Conector en línea de 2 pines
4		Conectores de tornillo de 2 pines
1		Conector DB9 hembra para impreso
1		Base para integrado de 28 pines, delgada
1		Base para integrado de 16 pines
1		Conector DB9 hembra con carcasa
1		Conector DB9 macho con carcasa
1m		Cable ribbon de 2 hilos
4		Tornillos de 1/2" X 1/8" con sus tuercas
1		Lámina de acrílico de 85mm X 70mm X 2,5mm
4		Separadores plásticos de 5mm



Paso 1

Instale y suelde las bases para los circuitos integrados IC1 e IC2.

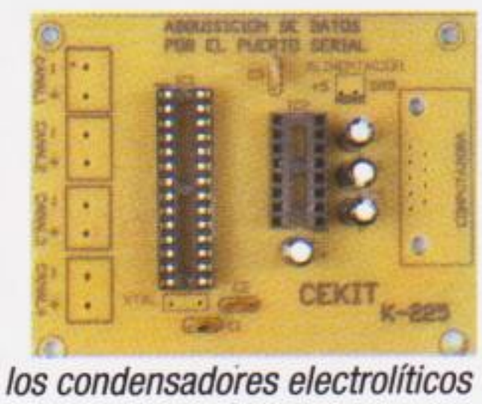
Figura 24.11



Paso 2

Luego instale y suelde los condensadores cerámicos del oscilador y del filtro de la fuente C1, C2 y C3.

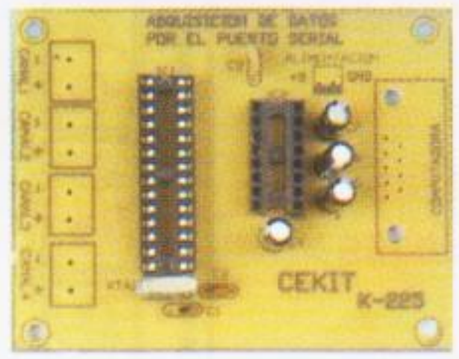
Figura 24.12



Paso 3

Instale y suelde los condensadores electrolíticos de 10µF C4, C5, C6 y C7.

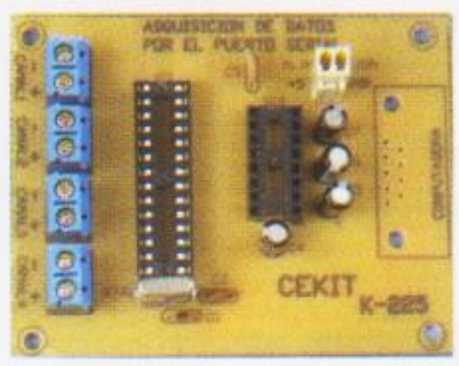
Figura 24.13



Paso 4

Instale y suelde el cristal de cuarzo XTAL, de 4MHz.

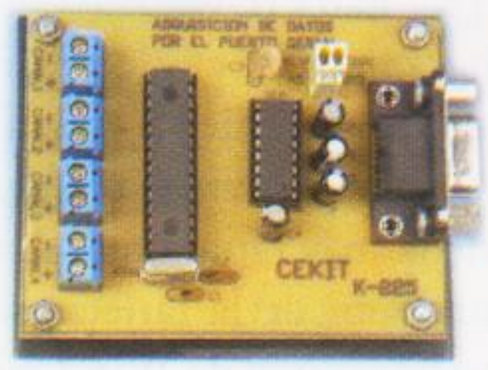
Figura 24.14



Paso 5

Instale y suelde el conector para la alimentación de +5VCC, y los conectores de tornillo de 2 pines, en las cuatro entradas (canal 1 a canal 4) .

Figura 24.15



Paso 6

Aspecto final de la tarjeta de adquisición de datos por el puerto serial K-225, luego de haber soldado el conector DB-9H para el puerto serial, y ubicados los circuitos integrados en sus respectivas posiciones.

Figura 24.16

Paso 7. Pruebas finales

Esta tarjeta debe alimentarse mediante un voltaje de CC de 5 V en el conector en línea de dos pines, ubicado en la parte superior derecha. Conecte adecuadamente la alimentación para evitar daños en los circuitos integrados. Por medio de un programa de prueba de PC, como el hyperterminal, haga una lectura de los datos que la tarjeta transmite de forma permanente.

PROYECTO 25

CERRADURA ELECTRÓNICA

Este proyecto permite tener en nuestro propio hogar u oficina un moderno sistema de control de entradas, antes exclusivo del sistema bancario. Con este sistema podemos decidir quien puede entrar y cuando, ya que es posible cambiar la "llave" (clave) en cualquier momento, a través del mismo teclado con el que escribimos el código de acceso.

Características

- Admite una clave de cuatro dígitos, entre los que se encuentran los números del 0 al 9, y las letras B, D, E y F.
- Estos cuatro dígitos posibilitan un total de 38.876 combinaciones diferentes.
- Fácil y segura modificación de la clave.
- Tamaño reducido, lo que permite una fácil instalación del sistema.

Funcionamiento

El sistema se puede dividir en dos partes: una lógica, y otra de potencia. En la primera se encuentra un microcontrolador PIC 16F84, el cual tiene como función recibir la clave proveniente del teclado y compararla con la que ha sido almacenada con anterioridad en la memoria; si éstas son iguales, el microcontrolador activa, a través del pin 6 (RB0), el circuito de la cantonera o cerradura eléctrica con el cual se puede abrir la puerta.

En la **figura 25.1** tenemos un diagrama de bloques genérico donde se puede apreciar la estructura del proyecto, en él se puede distinguir el microcontrolador PIC 16F84 que es el elemento central de procesamiento, al cual se conecta como dispositivo de entrada un teclado hexadecimal tipo membrana y, como dispositivos de salida, el circuito que maneja el cierre de la cantonera y el circuito que genera una señal audible por medio de un zumbador.

Dentro del código del programa se ha dispuesto el almacenamiento inicial de una clave de acceso en cuatro posiciones de la memoria EPROM de datos del microcontrolador. Es de anotar que gracias a la presencia de este tipo de memoria en el microcontrolador, nos ahorra tener que incluir una memoria serial externa, como una 24CXX o una 93XXX. El uso de estas memorias se justifica en casos donde debamos alamcenar gran cantidad de datos, como en un sistema de instrumentación. Este caso lo abordaremos en nuestro siguiente proyecto titulado «Datalogger de 12 bits».

El microcontrolador tiene como función recibir la clave proveniente del teclado y compararla con la que se ha almacenado; si éstas son iguales, el microcontrolador activa, a través del pin 6, el circuito de la cantonera, con lo cual se puede abrir la puerta.

Es claro que para lograr que la cerradura se abra, es necesario escribir en el teclado la clave correcta, ya sea la que se graba inicialmente en la memoria o la que el usuario haya grabado, puesto que es posible modificar la clave inicial desde el teclado. Cada vez que se digite cada uno de los números que conforman la clave, una señal audible indica su validez.

Una vez introducida la clave, el usuario debe pulsar la tecla A (Abrir); si la clave introducida es la correcta, la cantonera se abrirá durante

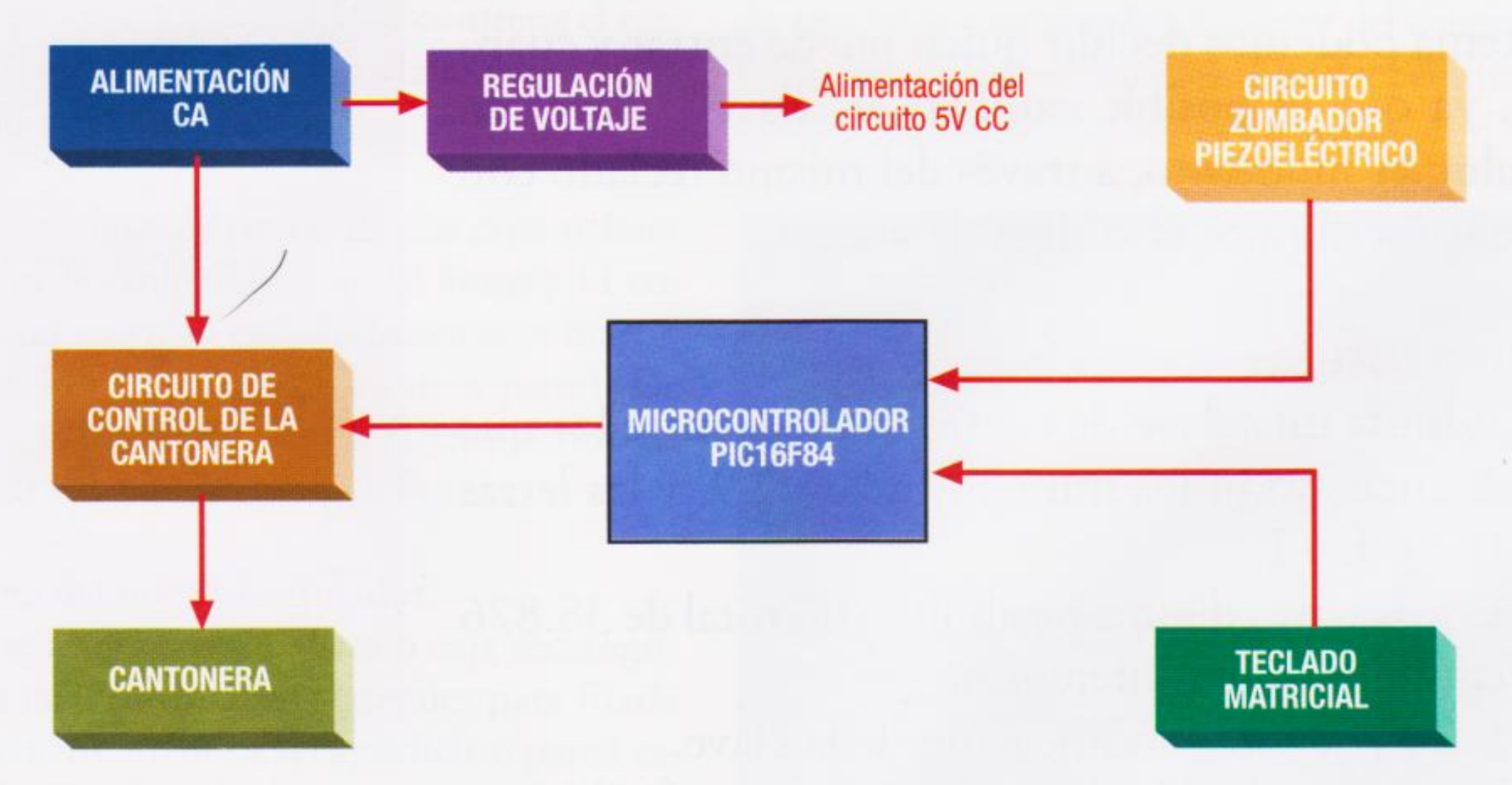


Figura 25.1 *Diagrama de bloques del proyecto*

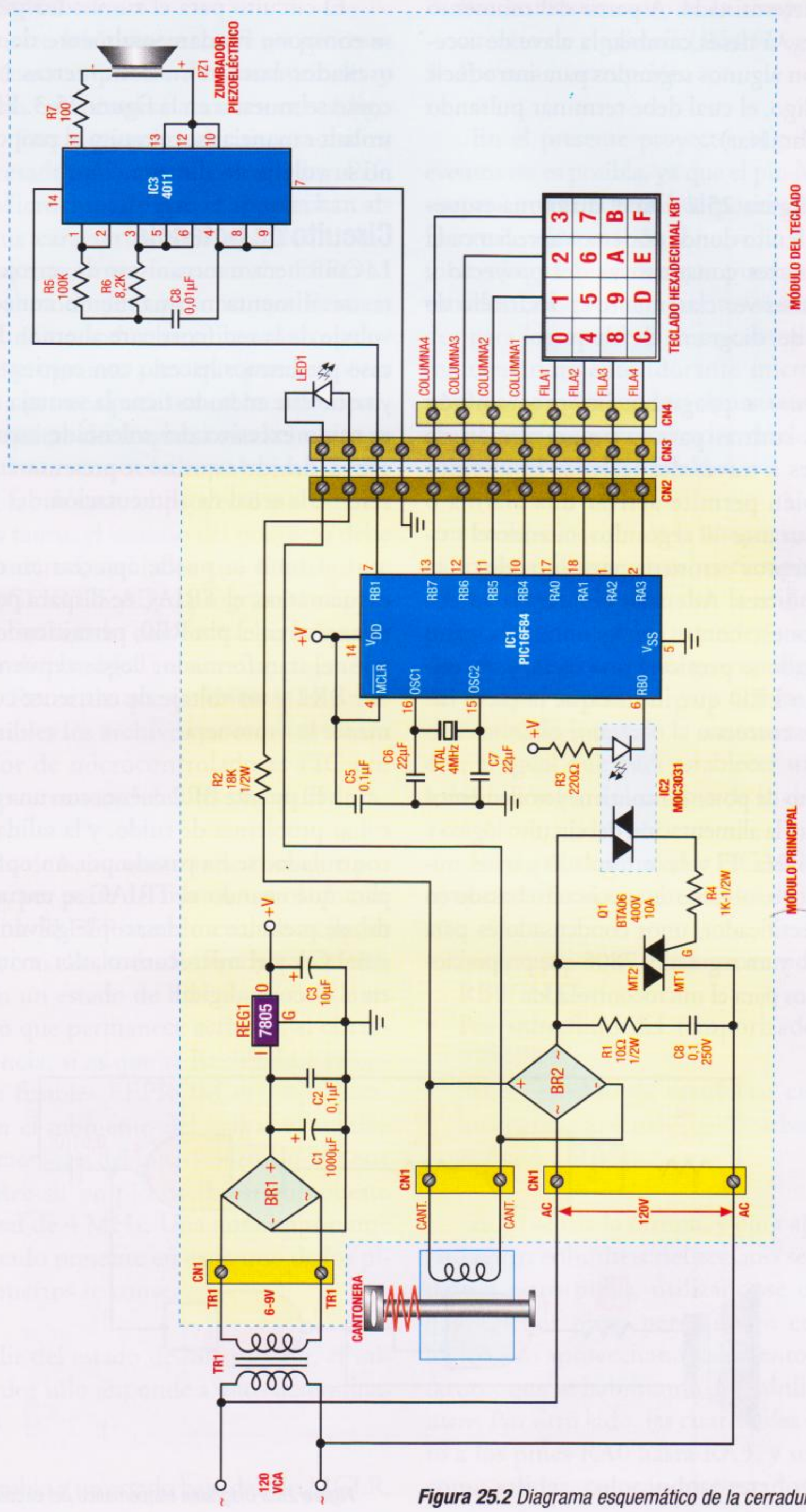


Figura 25.2 *Diagrama esquemático de la cerradura*

un tiempo determinado. A partir del momento de la apertura, si desea cambiar la clave de acceso, cuenta con algunos segundos para introducir el nuevo código, el cual debe terminar pulsando la tecla C (Cambiar).

En la **figura 25.2** está el diagrama esquemático del circuito donde podemos apreciar cada una de las partes constitutivas del proyecto y donde podemos ver claramente el desarrollo de los circuitos del diagrama de bloques.

El *software* o programa interno también contiene las rutinas para la programación de nuevas claves a través del teclado. El circuito lógico también permite activar una alarma o zumbador durante 40 segundos, cuando el número de intentos erróneos para introducir la clave llegue a tres. Además, el sistema ofrece otras funciones, como son generar un tono audible cuando se presiona una tecla, y encender un diodo LED que indica que la clave introducida es correcta.

El circuito de potencia contiene los elementos necesarios para la alimentación del circuito lógico y el de la cantonera. El voltaje regulado para el microcontrolador se obtiene de un circuito basado en un puente rectificador, unos condensadores para filtrar la señal, y un regulador 7805 que proporciona los 5 voltios para el microcontrolador.

El circuito para el zumbador piezoeléctrico se compone fundamentalmente de un circuito oscilador basado en compuertas NAND, tal como se muestra en la **figura 25.3**. El microcontrolador maneja este circuito al proporcionarle o no su voltaje de alimentación.

Circuito de potencia

La cantonera, o mecanismo de cerrojo de la puerta, se alimenta normalmente conectándola al voltaje de la red (corriente alterna). Pero en este caso preferimos hacerlo con corriente continua, ya que este método tiene la ventaja de eliminar el ruido excesivo del solenoide cuando éste se activa, debido a que no se presentarán cruces por cero de la señal de alimentación.

Como se puede apreciar en el diagrama esquemático, el TRIAC se dispara por medio de una señal en el pin RB0, permitiendo que el voltaje del transformador llegue al puente rectificador BR2 y un voltaje de corriente continua alimente la cantonera.

El puente BR2 cuenta con una red RC para evitar problemas de ruido, y la salida del microcontrolador se ha pasado por un optoacoplador para que cuando el TRIAC se encuentre cerrado, se presente un desacople galvánico entre la señal CA y el microcontrolador, evitando daños en el circuito digital.

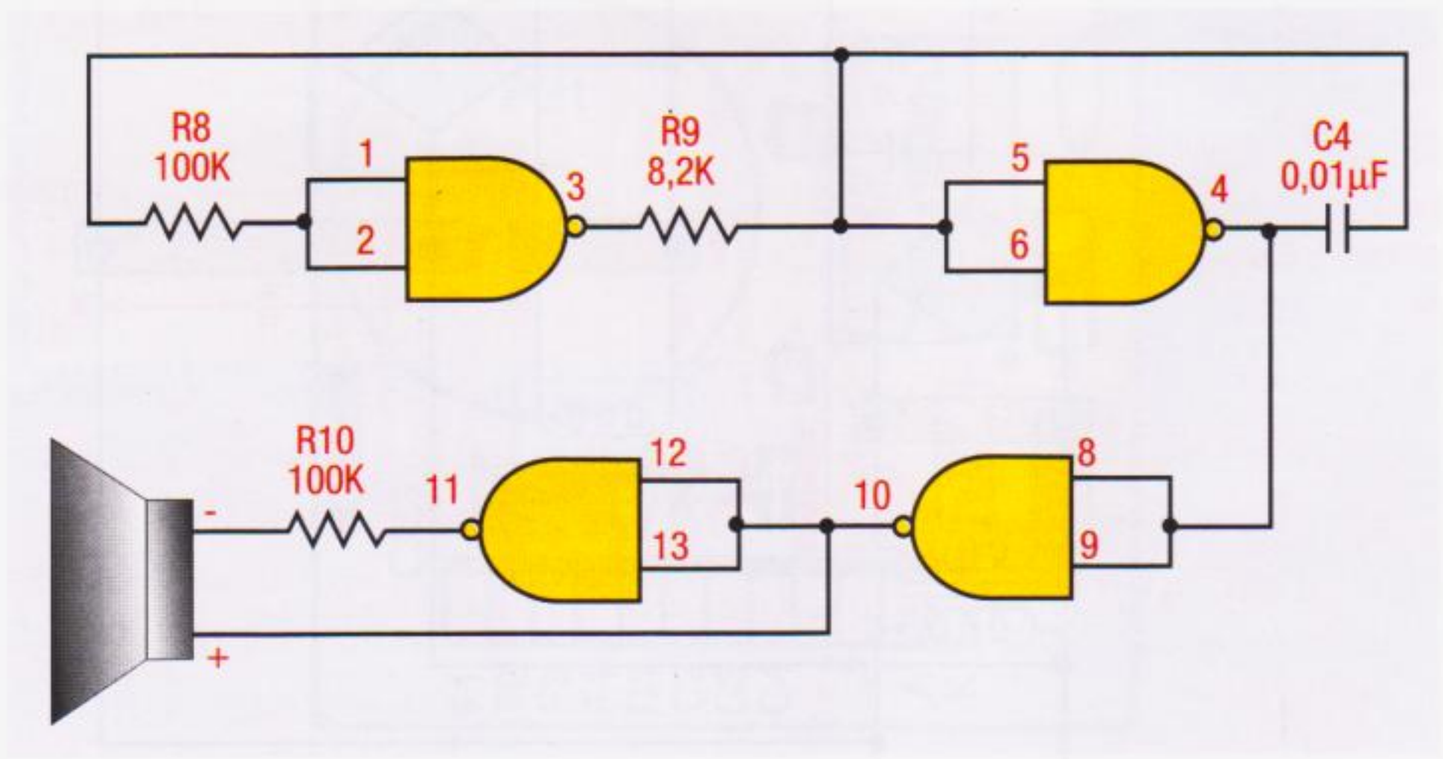


Figura 25.3 *Diagrama esquemático del circuito del zumbador*

Circuito lógico

Está compuesto básicamente, por un procesador que ejecuta una serie de instrucciones y una memoria en la cual se almacena el código del sistema. Para el presente circuito, se utiliza como procesador un microcontrolador PIC 16F84 de Microchip; de éste se aprovechan algunas de sus características especiales, como son las interrupciones, la memoria EPROM de datos, y el modo de bajo consumo. Recuerde que éste, como todos los microcontroladores, se debe programar para que pueda operar de acuerdo a la tarea para la cual ha sido destinado. Esto quiere decir que si en el circuito se conecta un microcontrolador tal como lo suministra el fabricante, el proyecto no funcionará; por lo tanto, el usuario del proyecto debe adquirir el microcontrolador con el fabricante de los kits (CEKIT S.A.), o programar su propio circuito integrado con los archivos que se encuentran disponibles en nuestra página en la internet. Recuerde que también se encuentran disponibles los archivos para construir un programador de microcontroladores PIC por medio del puerto serial de una computadora.

El modo de bajo consumo se consigue por la ejecución, por parte del microcontrolador, de la instrucción SLEEP, ante lo cual el oscilador se detiene y, con ello, todas las funciones de éste, entrando en un estado de hibernación. El único elemento que permanece activo es el circuito de vigilancia, si es que se ha dejado esta opción en los fusibles EEPROM del microcontrolador (en el momento del grabar el archivo HEX en la memoria del microcontrolador) ; este circuito posee su propio oscilador compuesto por un cristal de 4 MHz. Una nota importante es que el estado presente en cada uno de los pines de los puertos se conserva.

Para salir del estado de hibernación, el microcontrolador sólo responde a unos determinados eventos:

1- Por el cambio a un estado bajo del pin MCLR.
2- Por la terminación de la temporización del circuito de vigilancia (*WatchDog Timer*).
3- Por una interrupción.

En el presente proyecto, el primero de los eventos no es posible, ya que el pin MCLR se encuentra unido al terminal positivo (+5V) de la fuente de alimentación y no se ha dejado disponible ningun pulsador que pueda manipular el usuario. Los otros dos eventos sí son aprovechados para lograr que el microcontrolador atienda momentáneamente (durante microsegundos o unos pocos milisegundos) algunas condiciones externas, tales como la activación de elementos tales como una bocina o un relé.

La terminación de la temporización del circuito de vigilancia posee un tiempo nominal de 18 milisegundos, pero, con la utilización de un preescalador es posible extenderlo a 2 ó 3 segundos. Aquí se utiliza para generar algunas rutinas de tiempo, necesarias para activar durante un determinado intervalo la cantonera o el zumbador, al igual que para establecer un tiempo de habilitación para el cambio de la clave.

En el microcontrolador PIC 16F84 se poseen, entre otras, las siguientes fuentes de interrupción:

- Interrupción externa proveniente del pin RB0/INT
- Por sobreflujo del temporizador/contador TIMER0
- Por el cambio de estado en cualquiera de los cuatro pines más significativos del puerto B (RB4 a RB7).

Aquí se usa la última, y para aprovecharla, las cuatro columnas del teclado se han conectado a estos pines, utilizándose como entradas. Dichos pines permanecen en un estado lógico alto aprovechando elementos *pull-up* internos, que se habilitan o deshabilitan por *software*. Por otro lado, las cuatro filas se han puesto a los pines RA0 hasta RA3, y se configuran como salidas, colocándose estados lógicos ba-

jos en ellos con antelación a la ejecución de la instrucción SLEEP.

Cuando se pulsa cualquiera de los elementos del teclado, las columnas, que normalmente se encuentran en un estado alto (debido a los elementos *pull-up* internos habilitados), toman el estado lógico bajo (debido a los "ceros" que se encuentran en las columnas). Ésta es una de las situaciones que "despiertan" al microcontrolador, y, opcionalmente generan una interrupción (si es que éstas se encuentran habilitadas).

El código o clave del sistema se almacena en la memoria EPROM de datos.

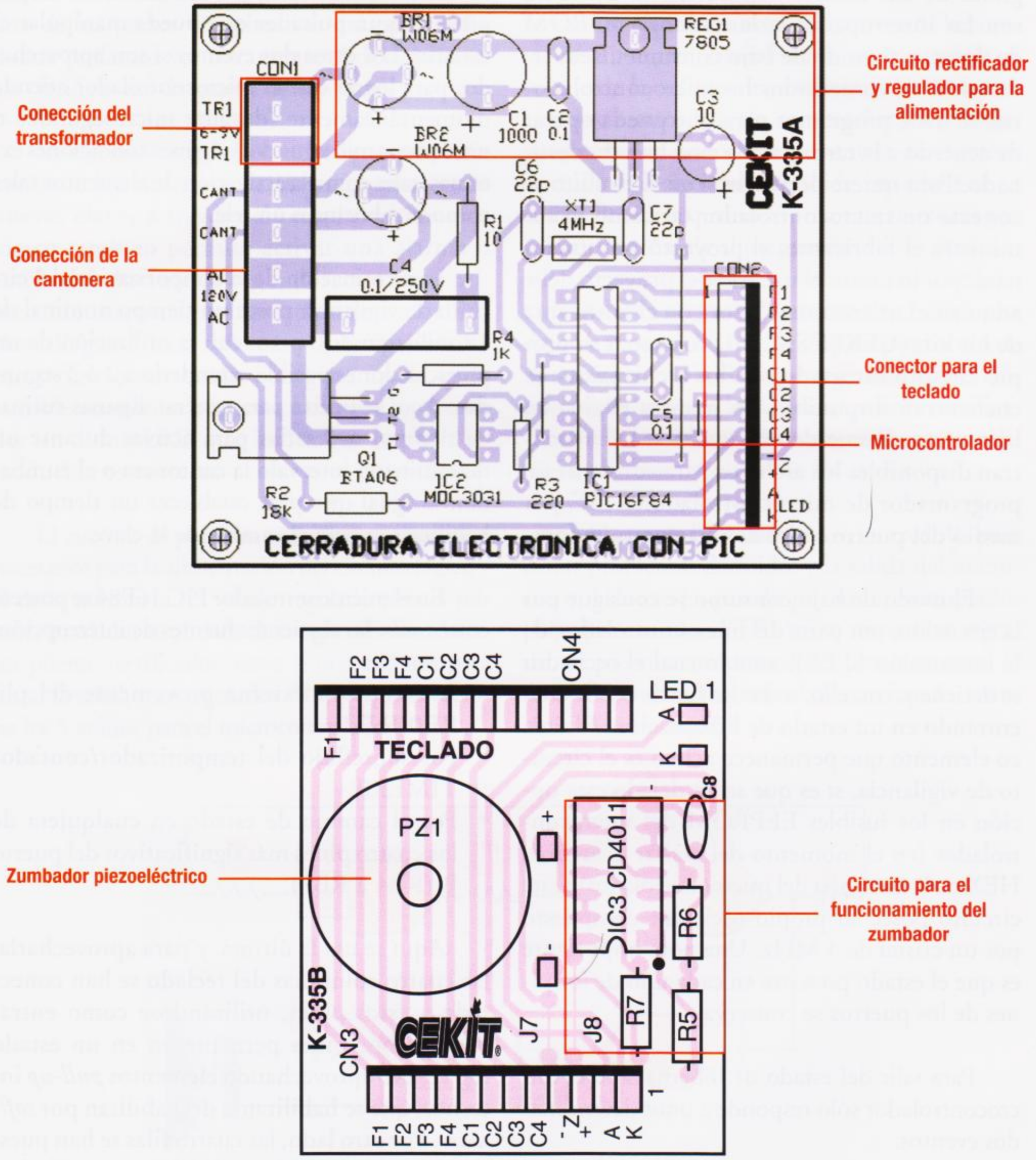


Figura 25.4 Guía de ensamblaje de la cerradura

Ensamblaje

Gracias a la tecnología de los microcontroladores, la construcción de dispositivos electrónicos se torna muy simple, pues la cantidad de componentes se reduce a menos de la mitad de lo que antes se requería para un diseño similar; además, el tiempo de ensamblaje de los circuitos también disminuye.

El cuidado en estos casos se centra en la manipulación del microcontrolador, ya que éste es sensible a la electricidad estática; por tanto, es el componente que se debe instalar en último término y sólo después de haber comprobado que la polaridad de la fuente de alimentación es correcta.

Inicialmente, instale en la tarjeta los puentes de alambre, recorte las partes sobrantes y suelde. Continúe con el montaje de las resistencias, los diodos, las bases para los circuitos integrados, los conectores, los condensadores y demás elementos. En las guías de ensamblaje de la **figura 25.4**, puede observar la distribución de los componentes.

Cuando haya montado completamente los circuitos impresos, proceda a comprobar su correcto funcionamiento, antes de instalarlos en el *chasis*. En primera instancia se deben comprobar los voltajes de alimentación, para ello conecte la alimentación, pero sin instalar todavía el microcontrolador. Con un voltímetro en la escala de voltios de CC, mida el voltaje a la salida del rectificador, éste debe estar en 5 voltios.

Introducción de la clave

Para lograr la apertura del sistema, el usuario tiene que escribir en el teclado la clave correcta, introduciendo secuencialmente los cuatro dígitos adecuados. Para iniciar, el sistema viene programado previamente con el código 0123, **figura 25.5**.

Al escribir cada uno de los dígitos que conforman la clave, una señal sonora avisa la aceptación de éstos; si se equivoca al introducir algún dígito, sólo tiene que escribir de nuevo el código correcto.

El sistema se ha diseñado de tal forma que almacena los cuatro últimos dígitos introducidos desde el teclado; para que la clave sea validada, se hace necesario que el usuario pulse la tecla A (Abrir). Si los dígitos introducidos coinciden con la clave del sistema, la cantonera conectada al terminal de salida RB0 se activará durante aproximadamente 1,5 segundos, tiempo que se considera suficiente para los propósitos de apertura de una puerta, por ejemplo.

Cambio de clave

Para modificar la clave, sólo puede hacerlo después de una condición de apertura. A partir

Figura 25.5 *Interacción del usuario con la cerradura*

del momento en el cual se desactiva la carga, cuenta con ocho (8) segundos para introducir el nuevo número deseado; finalizado esto, debe pulsar la tecla C (Cambiar) e introducir de nuevo el código, como una verificación del número. Después de pulsar la tecla C se puede tomar hasta 30 segundos para escribir el código de verificación.

Si ambos códigos no coinciden, la operación es interrumpida y el código que queda vigente es el que se tenía antes. Si ambos códigos son iguales, el nuevo código se almacenará en la memoria del sistema. Después de dos segundos de pulsar la última tecla del código de verificación, presione la tecla de apertura: la cantonera deberá activarse, confirmando la validación de la nueva clave.

Instalación

Una vez que se haya comprobado el correcto funcionamiento de la cerradura (eléctrico y operativo), se procede a la instalación de las tarjetas dentro de las cajas metálicas que van incrustadas en la pared.

La caja que contiene el circuito de zumbador y el conector para el teclado, se instala por fuera y a la entrada del lugar que se quiere proteger; mientras que la caja que contiene el microcontrolador queda montada al interior (al otro lado de la pared).

La comunicación entre las dos cajas se hace por medio de cable ribbon de 12 líneas y 11 cm de longitud (si en su caso la distancia es mayor, puede utilizar cable telefónico de 6 pares). De otro lado, la instalación de las tarjetas dentro de las cajas se hace de la siguiente forma:

a - Tarjeta del microcontrolador.

Ésta se monta en un *chasis* o caja rectangular, la cual posee aletas laterales para fijarla permanentemente a la superficie o pared escogida. Para instalar la tarjeta se utilizan tornillos de cabeza plana de 1/2" de longitud y separadores de plástico de 5mm. El *chasis* tiene perforaciones a través de las cuales se pasan los cables de alimentación, el de la cantonera, y el de comunicación con el teclado. En la **figura 25.6** se puede apreciar este *chasis*.

b - Tarjeta del teclado.

Inicialmente sobreponga el *chasis* del teclado sobre la superficie escogida, luego marque con un lápiz de punta fina los puntos donde van los tornillos de fijación. Retire la caja y perfore orificios de tamaño suficiente para que pueda instalar los tacos (chazos) para los tornillos.

Ahora instale la tarjeta dentro del *chasis* usando tornillos de cabeza plana de 1/2" por 1/8", y separadores de plástico de 5mm. Proceda a conectar el cable del teclado al conector CN2. Terminada esta operación puede instalar el conjunto en el lugar escogido, con lo cual sólo falta conectar el teclado y cerrar la caja definitivamente.

El teclado se asegura sobre la tapa gracias a una sustancia adhesiva que normalmente se encuentra protegida con papel siliconado. Los terminales de conexión del teclado se pasan a través de una ranura localizada a la altura del conector CN4, esto facilita el ensamblaje final y permite el cambio del teclado cuando se requiera.

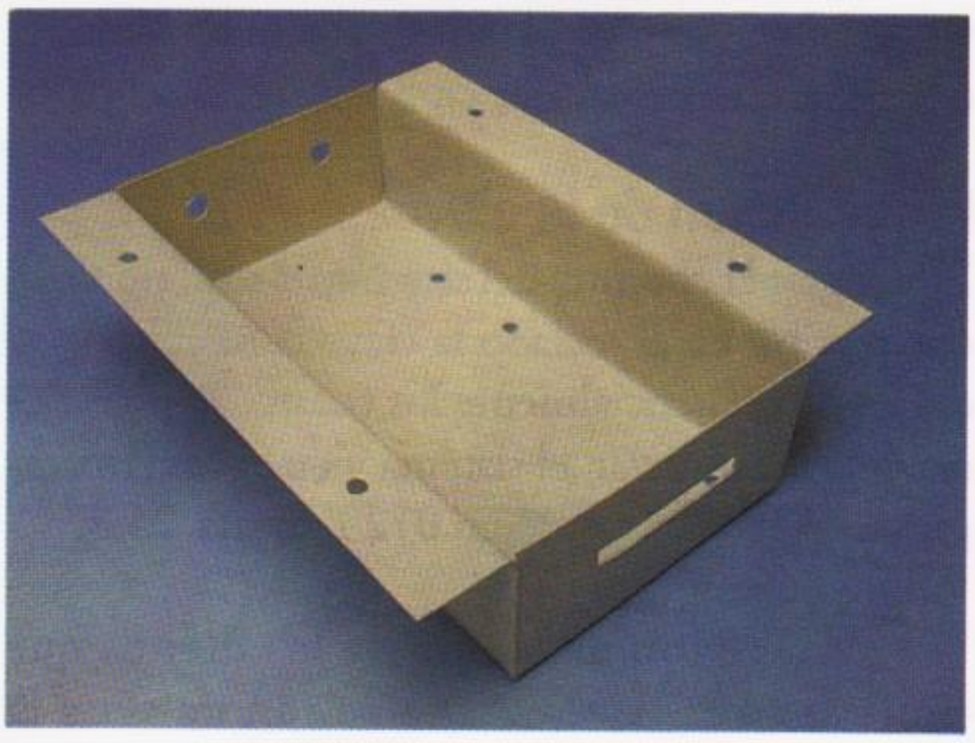


Figura 25.6 *Chasis del proyecto*

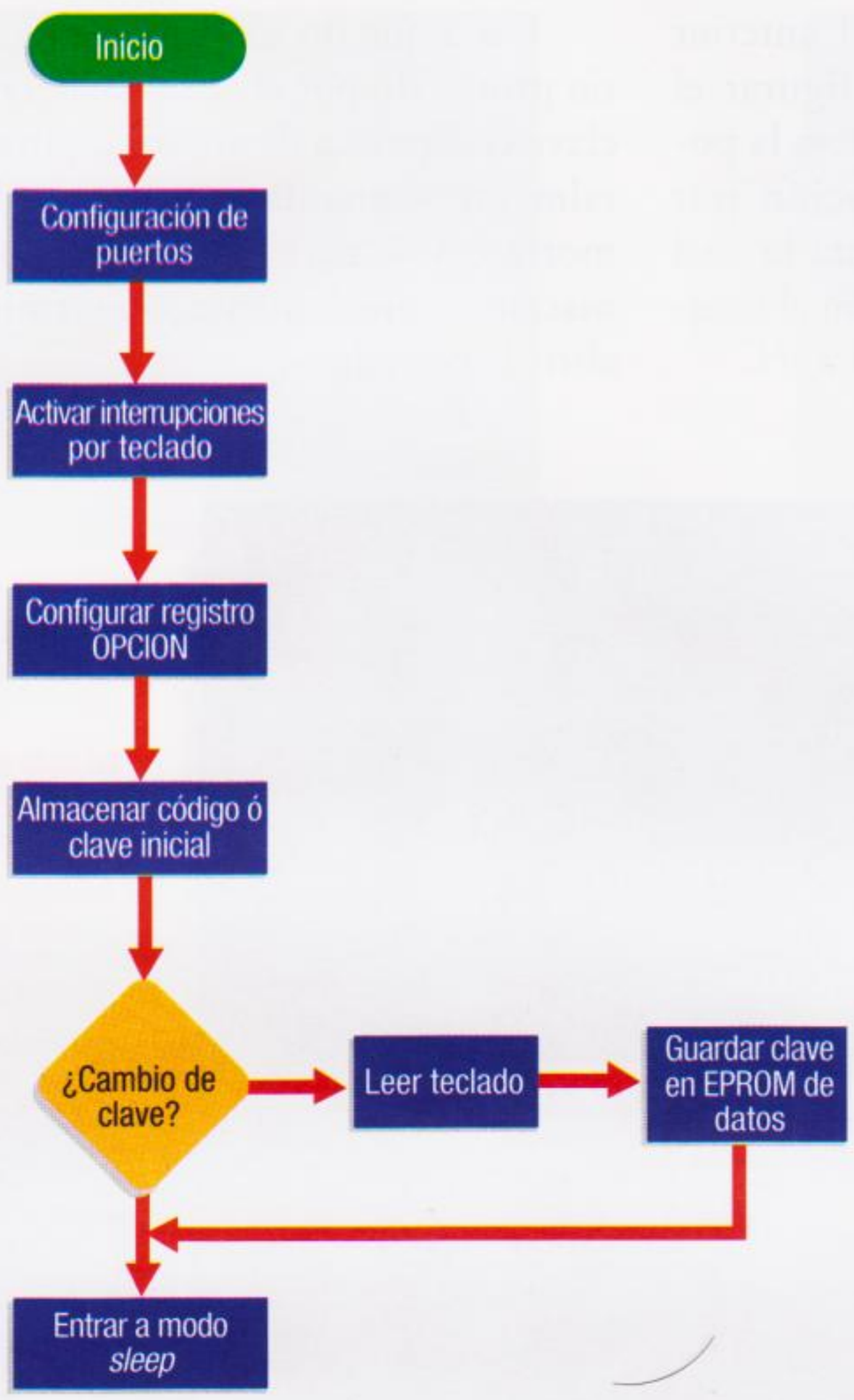


Figura 25.7 *Diagrama de flujo de la cerradura*

Descripción del programa

En la **figura** 25.7 podemos apreciar el diagrama de flujo general del proyecto y el código fuente se encuentra disponible en nuestra página en la internet en el enlace Curso Práctico sobre Microcontroladores; ahora nos limitaremos a una descripción muy general del código fuente escrito en lenguaje ensamblador y sus partes fundamentales.

El esquema general es el siguiente: dentro del código del programa se almacena una clave de acceso inicial en cuatro posiciones de la memoria EPROM de datos. Recordemos que la EPROM de datos es un bloque de memoria independiente de la memoria de programa y de la memoria de datos.

Se trata de un tipo de memoria que obviamente posee un conjunto de direcciones que se pueden acceder a través de registros especiales y sobre las cuales podemos almacenar datos que pueden modificarse estando dentro del funcionamiento normal del circuito.

Por su naturaleza EPROM, los datos se mantendrán almacenados sin importar que no haya alimentación en el circuito, lo que permite que la clave permanezca almacenada indefinidamente, a menos que se vuelvan a grabar las mismas posiciones.

La posibilidad de contar con la EPROM de datos en el microcontrolador PIC16F84, nos ahorra el trabajo de tener que conectar una memoria serial para almacenar la clave y de alguna manera la necesidad de contar con un dispositivo programador de memorias seriales.

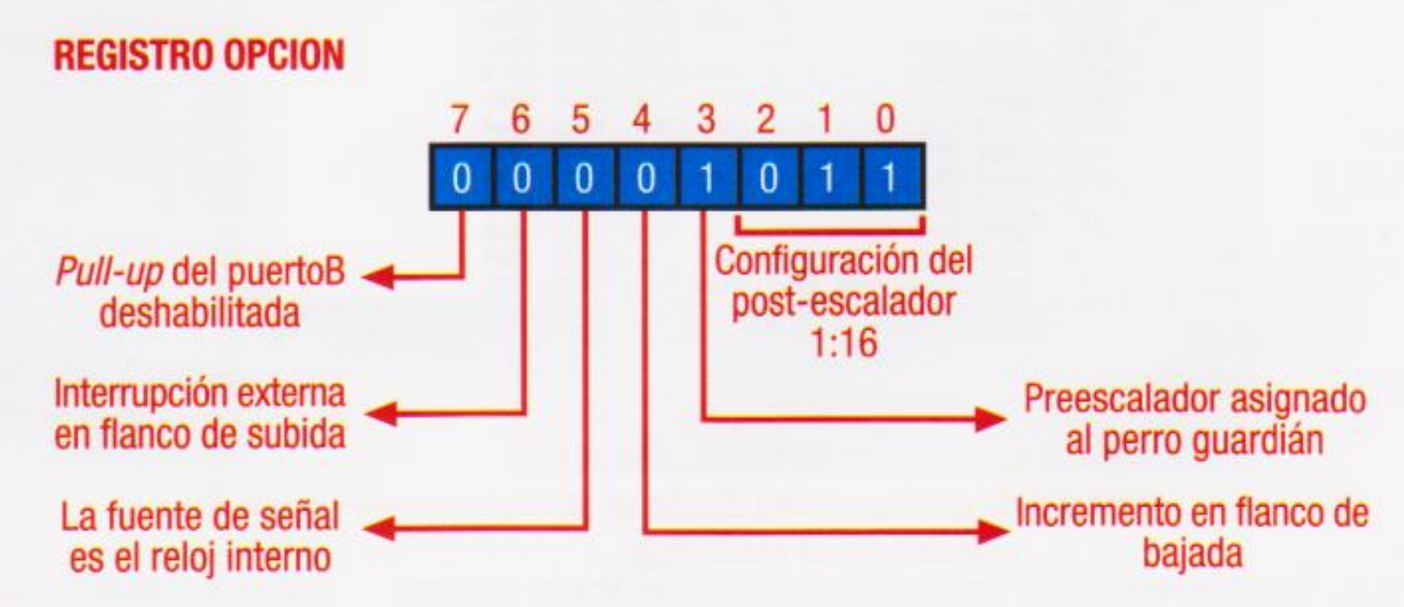


Figura 25.8 *Configuración del registro OPCION*

Como se puede observar en el anterior diagrama de flujo, después de configurar el funcionamiento de los puertos, se activa la posibilidad de generar una interrupción por cambio de nivel en el puerto B, para lo cual debe activar los *bits* correspondientes en el registro de control de interrupciones INTCON.

Cada que un usuario quiere acceder al sitio protegido por el sistema, debe escribir una clave compuesta de cuatro dígitos que temporalmente se guarda en cuatro registros de memoria RAM, para compararla con la clave almacenada previamente y determinar si se debe abrir la cerradura.

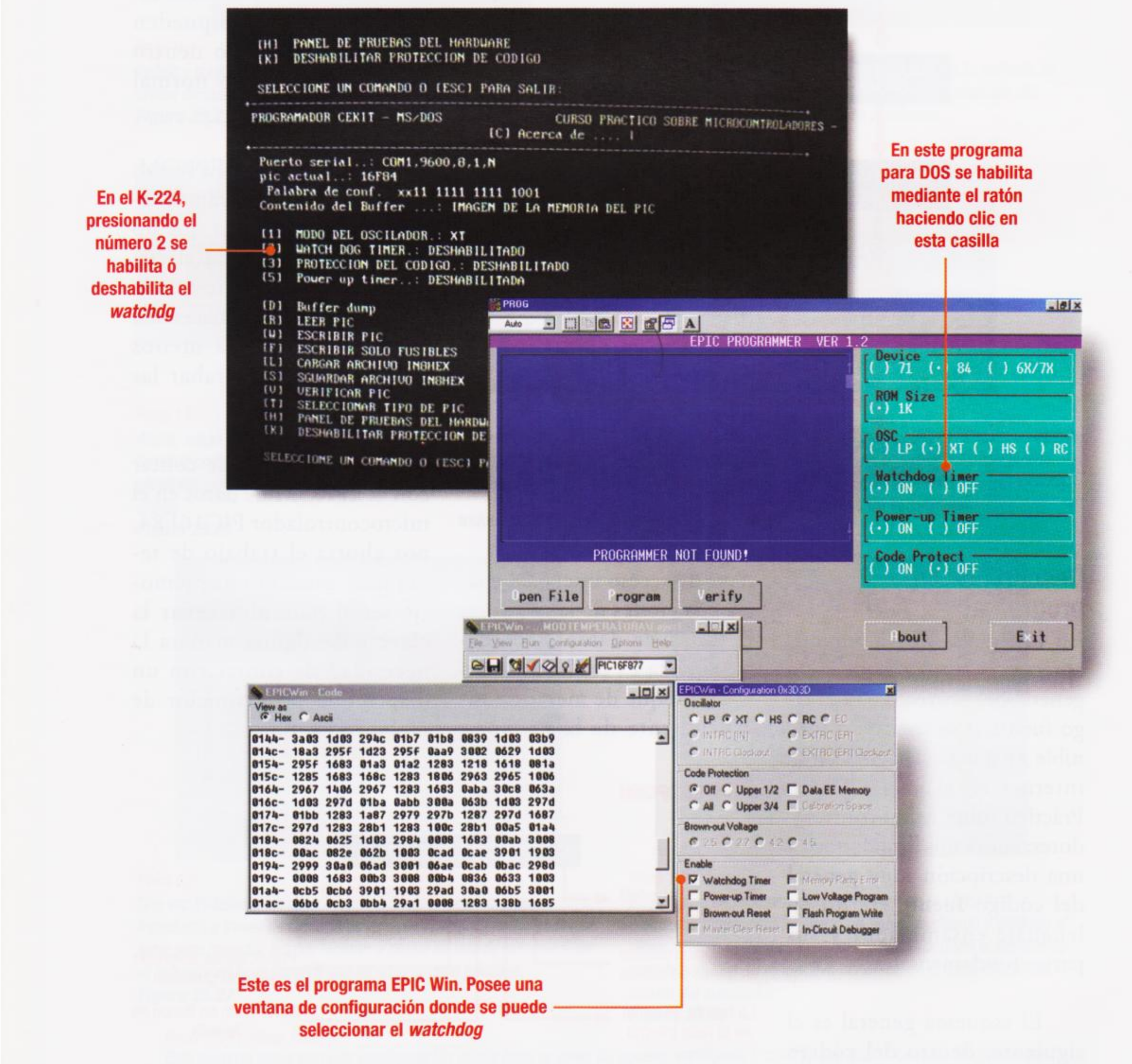


Figura 25.9 *Habilitación del WATCHDOG*

La configuración del registro OPCION es como se muestra en la **figura 25.8**. Principalmente podemos notar que en este proyecto no se utilizó el TIMER0 para generar interrupciones, ni como temporizador o contador; por el contrario, se utilizó el *WATCHDOG* razón por la cual se le asignó el preescalador (ó post-escalador).

La posibilidad de trabajar con el *WATCHDOG* debe estar habilitada al momento de grabar o «quemar» el microcontrolador; para ello todos los sistemas de programación deben poseer dentro de la interfaz de usuario, la posibilidad de activar esta opción, **figura 25.9**.

Siguiendo el orden dispuesto en el diagrama de bloques se debe proceder a almacenar en la EPROM de datos la clave de acceso inicial.

Es importante aclarar que la clave inicial puede ser modificada; de esta manera le garantiza total seguridad en el momento que instale el sistema en su hogar u oficina. Cuando se desea cambiar la clave, el programa lee el teclado y guarda el dato leído en la EPROM, finalmente entra en modo de bajo consumo o modo SLEEP.

La lectura normal del teclado sólo se desarrolla dentro de la subrutina o servicio de interrupción, no dentro del cuerpo principal del programa. Sólo tiene lugar cuando se detecta un cambio de nivel en el puerto B y se genera la interrupción por teclado.

En lecciones anteriores ya hemos analizado detalladamente cómo se efectúa el barrido y todo el procedimiento de lectura de las columnas y las filas del teclado para deducir acertadamente el valor de la tecla que se presiona en un momento dado.

Para habilitar el funcionamiento de esta interrupción se debe configurar adecuadamente el registro INTCON, como se muestra en la **figura 25.10**. Podemos observar cómo se deben activar los *bits* tres y siete de este registro, que corresponden al habilitador de interrupciones por cambio en el puerto B (RBIE) y al habilitador global de interrupciones (GIE). En la porción de código de la **figura 25.11** podemos apreciar la configuración del registro OPCION, INTCON, de los puertos, y el momento en que se pone el microcontrolador en modo SLEEP.

Podemos observar que el programa se encierra en un pequeño lazo en el cual se interroga permanentemente por el *bit* cuatro del registro de estado, **figura 25.12**, el cual es una bandera que nos indica cuando ha terminado de temporizar el *WATCHDOG*. Mientras no haya terminado este proceso, el microcontrolador no saldrá de su estado de bajo consumo. Cada que sale del modo SLEEP el programa hace un chequeo de todos los eventos que tuvieron lugar mientras se encontraba "durmiendo" y que se deben a las acciones efectuadas durante el servicio de interrupción.

Todos estos eventos se agrupan en un registro bandera denominado *flags1*, y, de acuerdo al cambio de los *bits* de este registro, el programa salta a distintas subrutinas que llevan a cabo las acciones necesarias para cumplir con los objetivos propuestos en el sistema, como por ejemplo, determinar cuando se quiere cambiar la clave, o bloquear la cerradura en caso de que se hayan intentado cuatro o más claves incorrectas.

El servicio de interrupción por cambio en el puerto B, algunas veces denominado interrupción

Figura 25.10 *Configuración del registro INTCON*

```
SIGUE     BSF      STA,5              ;SELECCIONAR PÁGINA 1
          CLRW                        ;LIMPIAR REGISTRO W
          MOVWF    TRISA              ;CONFIGURAR PUERTO A COMO SALIDA
          MOVLW    B'11111100'        ;MEMORIA COMO SALIDA
          MOVWF    TRISB              ;CONFIGURAR PUERTO B
          BCF      STA,5              ;IR A P†GINA 0
          MOVF     JUFE,W             ;LLEVAR CONTENIDO PPAL A W
          MOVWF    PORT_B             ;PARA LLEVAR AL PUERTO B
          CLRF     INTCON             ;LIMPIAR REGISTRO DE INTERRUPCIONES
          BSF      INTCON,3           ;ACTIVAR INTERRUPCIÓN POR TECLADO
          BSF      INTCON,7           ;ACTIVAR LAS INTERRUPCIONES
          BTFSS    STA,TO             ;¿RESET POR TEMPORIZADOR?
          GOTO     DORM

DORSIL    BCF      JUFE,SNA           ;APAGAR EL ZUMBADOR
          MOVF     JUFE,W             ;LLEVAR CONTENIDO PPAL A W
          MOVWF    PORT_B             ;PARA EL PUERTO B

DORM      MOVLW    0                  ;LIMPIAR REGISTRO W
          MOVWF    PORT_A             ;PARA LLEVAR A FILAS TECLADO
ZZZZ      BSF      STA,5              ;PÁGINA 1
          CLRWDT                      ;LIMPIAR CIRCUITO DE VIGILANCIA
          MOVLW    B'00001011'        ;PULL-UPS, WDT, 1/8
          MOVWF    OPCION             ;
          BCF      STA,5              ;PÁGINA 0
DORMITA   SLEEP                       ;DORMITAR
          NOP                         ;
          NOP
WAKE      BTFSS    STA,TO             ;AVERIGUAR SI FIN WATCHDOG
          GOTO     WATCH              ;SI ES ASÍ, IR A TRATAR
          GOTO     DORMITA

WATCH     BTFSS    FLAGS1,TEMPO       ;BANDERA DE TIEMPO
          GOTO     DORM               ;SI NO ACTIVA, IR A DORMITAR
          DECFSZ   CUENTA             ;DECREMENTAR CUENTA
          GOTO     DORM               ;SI NO ES CERO, IR A DORMITAR
          BCF      FLAGS1,TEMPO       ;LIMPIAR BANDERA DE TIEMPO
          BTFSC    FLAGS1,CANTO       ;PREGUNTAR POR CANTONERA
          GOTO     CANTOS             ;SI ACTIVA, IR A TRATAMIENTO
          BTFSC    FLAGS1,CODE        ;¿BANDERA DE CAMBIO DE CÓDIGO?
          GOTO     DONES              ;SI ACTIVA, IR A TRATAMIENTO
          BTFSC    FLAGS1,SNA         ;¿BANDERA DE ALARMA?
          GOTO     ALAS               ;SI ACTIVA, TRATAR
          BTFSC    FLAGS1,CH1         ;¿BANDERA DE CAMBIO DE CÓDIGO?
          GOTO     CAMBIO1            ;SI ACTIVA, IR A TRATAMIENTO
          BTFSC    FLAGS1,CH2         ;¿BANDERA DE CAMBIO 2?
          GOTO     CAMBIO2            ;IR A TRATAR
          BTFSC    FLAGS1,ERRO        ;¿BANDERA DE ERROR?
          GOTO     ERROS              ;IR A TRATAMIENTO DE ERROR
          GOTO     DORM               ;NINGUNA DE LAS ANTERIORES, DORMITAR
```

Figura 25.11 *Porción del código fuente de la interrupción*

Figura 25.12 Registro de estado.

por teclado, es el que observamos en el diagrama de flujo de la **figura 25.13**. La primera acción que se ejecuta es la lectura de los cuatro dígitos entregados desde el teclado y su almacenamiento en registros en memoria RAM.

Uno por uno se comparan con los dígitos almacenados que corresponden a la clave correcta. Si son iguales, se genera la acción para abrir la cantonera; si la clave introducida no es la correcta, se incrementa un contador de intentos fallidos; si este contador llega a cuatro, se desactiva la cantonera en forma definitiva.

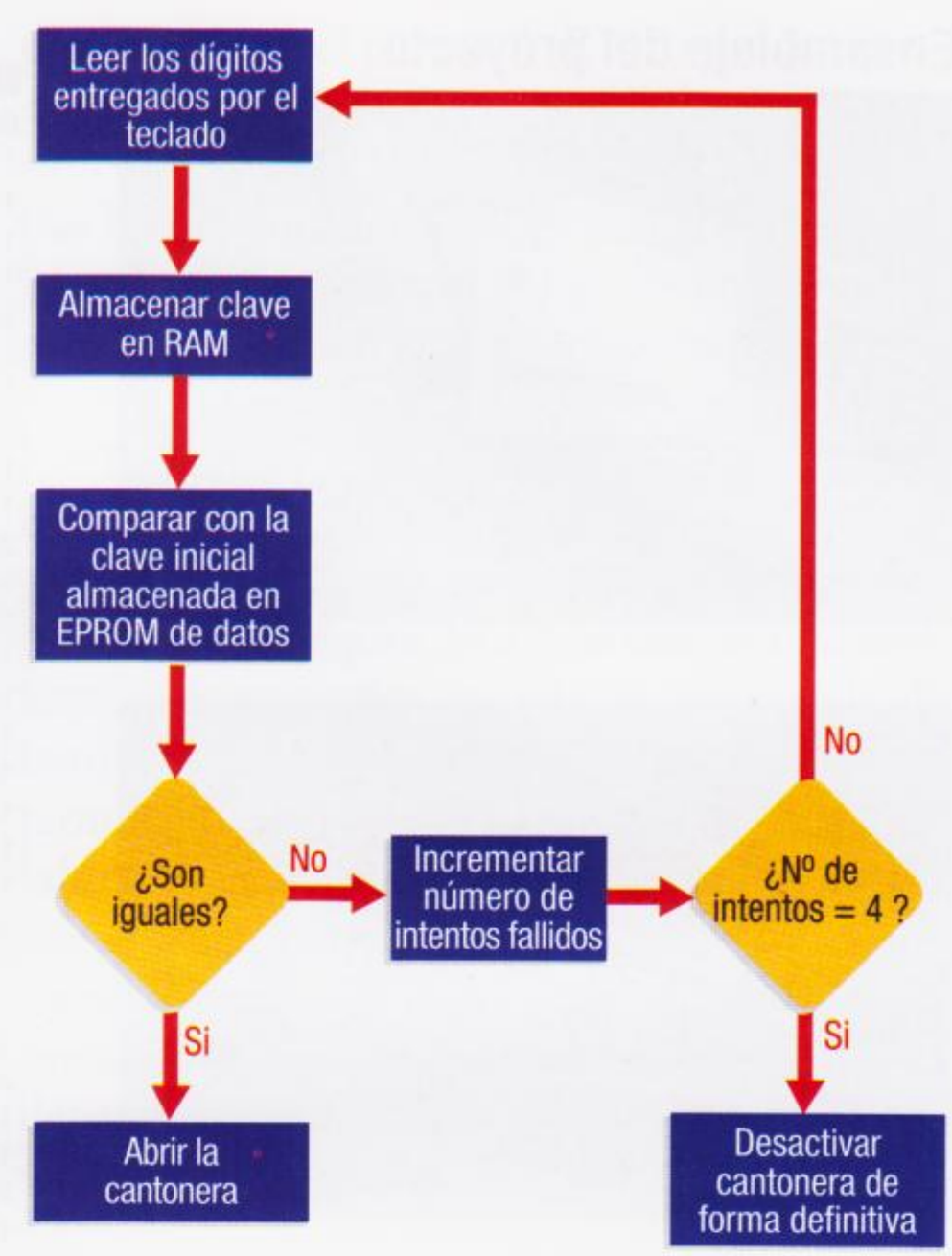


Figura 25.13 Diagrama de flujo del servicio de interrupción

En la **figura 25.15** observamos parte del código fuente del servicio de interrupción; podemos apreciar que entre las primeras acciones que se ejecutan está el borrado del perro guardián y el borrado de la bandera de interrupción por cambio en el puerto B (INTCON,RBIF) para permitir que el microcontrolador pueda generar una nueva interrupción y ejecutar de nuevo esta subrutina; posteriormente se llevan a cabo las demás acciones contempladas dentro del diagrama de flujo.

```
        ORG         0                   ;VECTOR DE RESET
        NOP
        NOP
        NOP
        GOTO        INICIO              ;VAYA A COMIENZO

        ORG         4                   ;VECTOR DE LA INTERRUPCIÓN
        NOP
        BSF         STA,5               ;SI NO ES ASÍ, ACTIVAR PÁGINA 1
        CLRWDT                          ;LIMPIAR CIRCUITO DE VIGILANCIA
        MOVLW       B'00001111'         ;SELECCIONAR WDT, 1/128
        MOVWF       OPCION              ;PARA REGISTRO OPTION
        BCF         STA,5               ;IR A PÁGINA 0
        BCF         INTCON,RBIF         ;LIMPIAR BANDERA INTERRUPCIÓN
...
                                        ; POR LECTURA DEL TECLADO
```

Figura 25.14 Porción de código de la interrupción

Ensamblaje del proyecto

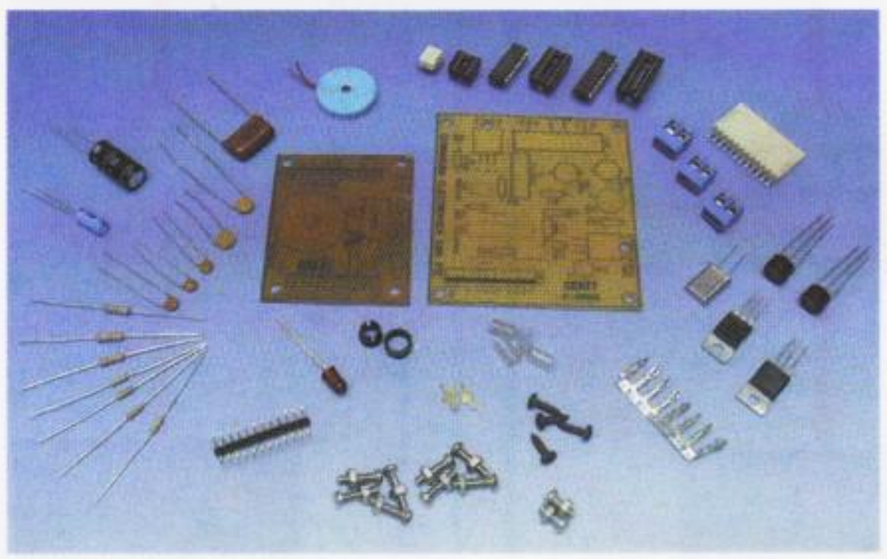
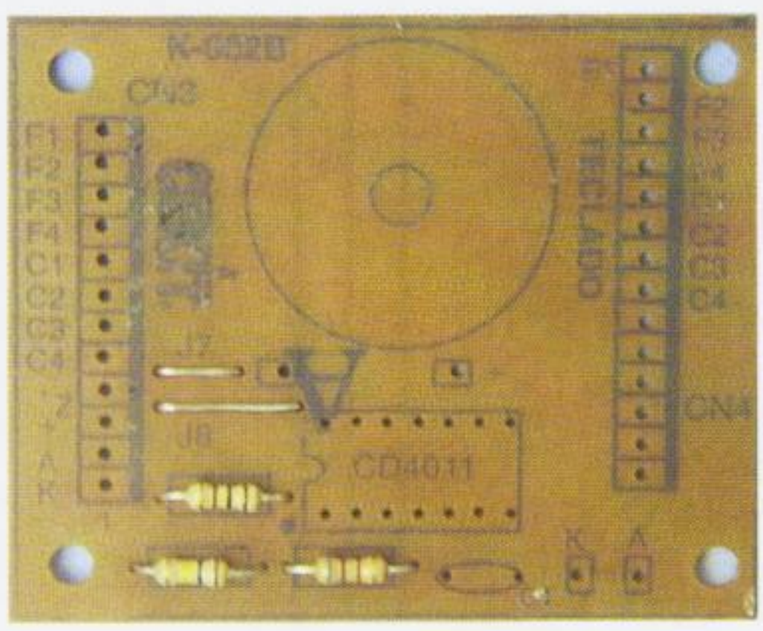


Figura 25.14 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar la cerradura electrónica CEKIT K-335.*

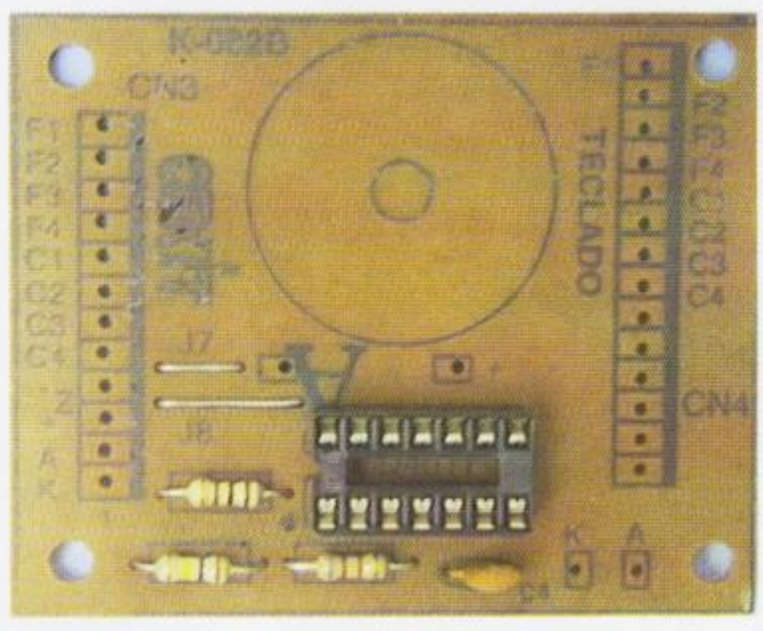
Este sencillo proyecto usted podrá utilizarlo como una cerradura electrónica en el interior de su vivienda, con ella podrá proteger el acceso a algunos lugares que pueden ser privados o deben mantenerse vigilados, como por ejemplo el armario, la caja fuerte, la puerta de la habitación, o el acceso a lugares especiales. Para su instalación se requiere de una cantonera de 110VCA y un adecuado acople mecánico con la chapa. Para el ensamblaje de las tarjetas de este proyecto tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

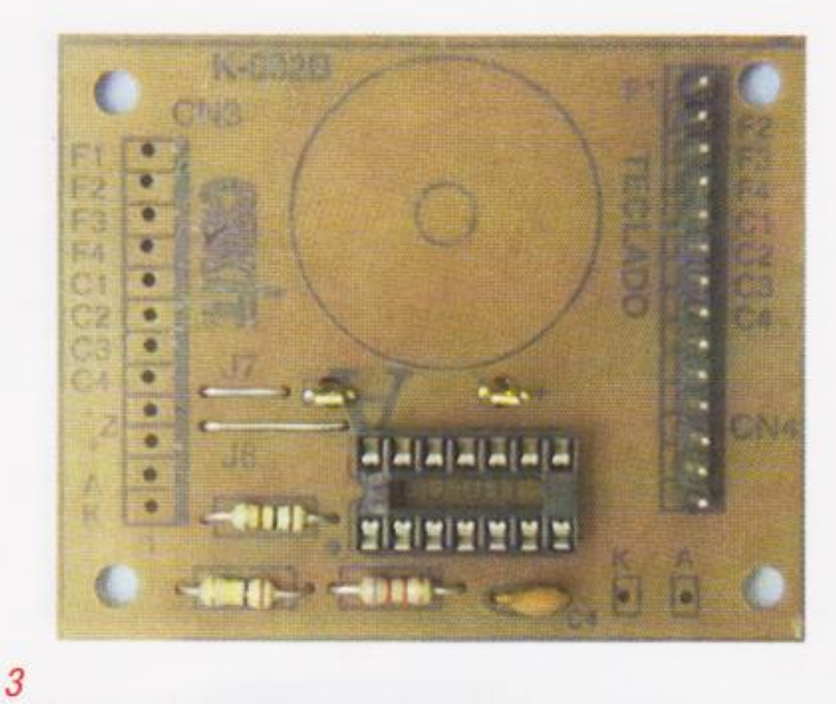
Cant.	Ref.	Descripción
1	IC1	Microcontrolador PIC16F84, programado
1	IC2	Circuto integrado optotriac MOC3031
1	IC3	Circuto integrado CD4011
1	REG1	Regulador fijo de +5V -1A, 7805
2	BR1 y BR2	Puentes rectificadores circulares de 1A, W06M
1	LED1	Diodo LED rojo de 5mm
1	C1	Condensador electrolítico de 1000uF - 25V
2	C2 y C5	Condensadores cerámicos de 0.1uF - 50V
1	C3	Condensador electrolítico de 10uF - 25V
1	C4	Condensador cerámico de 0.1uF - 250V
2	C6 y C7	Condensadores cerámicos de 22pF - 50V
1	C8	Condensador cerámico de 0.01uF - 50V
1	XT1	Cristal de quarzo de 4MHz
1	R1	Resistencia de 10 - 1/2W
1	R2	Resistencia de 18k - 1/2W
1	R3	Resistencia de 220 - 1/4W
1	R4	Resistencia de 1k - 1/2W
1	R5	Resistencia de 100k - 1/4W
1	R6	Resistencia de 8.2k - 1/4W
1	R7	Resistencia de 100 - 1/4W
1	Q1	Triac de 10A, BTA06-400, ó equivalente
1	TR1	Transformador de 9V-450mA (Magom M-504)
1	PZ1	Buzzer o parlante piezoeléctrico
1	KB1	Teclado matricial de membrana o teclas
2	K-335A y K-335B	Circuitos impresos CEKIT
1	Chasis K-335A	Chasis metálico para el circuito principal
1	Chasis K-335B	Chasis metálico para el teclado y el zumbador
3	CON1	Conectores de tornillo de 2 pines
1	CON2	Conector en línea de 12 pines M/H
1	CON4	Conector tipo cerca no polarizado de 14 pines
1		Conector ribbon hembra de 28 pines para el teclado
4		Espadines para circuito impreso
1		Base para circuito integrado de 6 pines
1		Base para circuito integrado de 18 pines
1		Base para circuito integrado de 14 pines
1		Porta LED de 5mm
2		Pasa cables de caucho
8		Tornillos de 1/2" X 1/8", con sus tuercas
2		Tornillos de 1/4" X 1/8", con sus tuercas
4		Tornillos golosos de 3mm para lámina
8		Separadores plásticos de 5mm
1m		Cable ribbon de 12 hilos
1m		Cable tipo vehículo No. 20AWG
1		Cable de potencia



Paso 1
Como primer paso se ensamblará la tarjeta del zumbador; para ello instale y suelde los puentes con alambre telefónico desnudo Nº 24AWG, y también las tres resistencias de 1/4 de vatio.
Figura 25.15



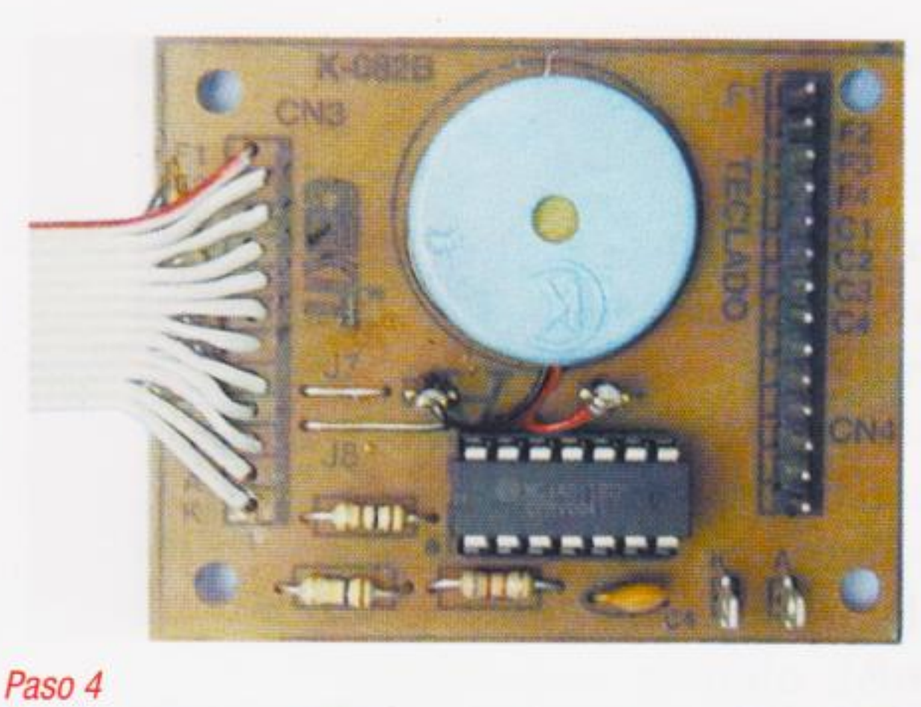
Paso 2
Instale y suelde la base para el circuito integrado CD4011 y el condensador cerámico.
Figura 25.16



Paso 3

Instale y suelde el conector macho tipo cerca no polarizado para conectar el teclado, y los dos espadines para el zumbador.

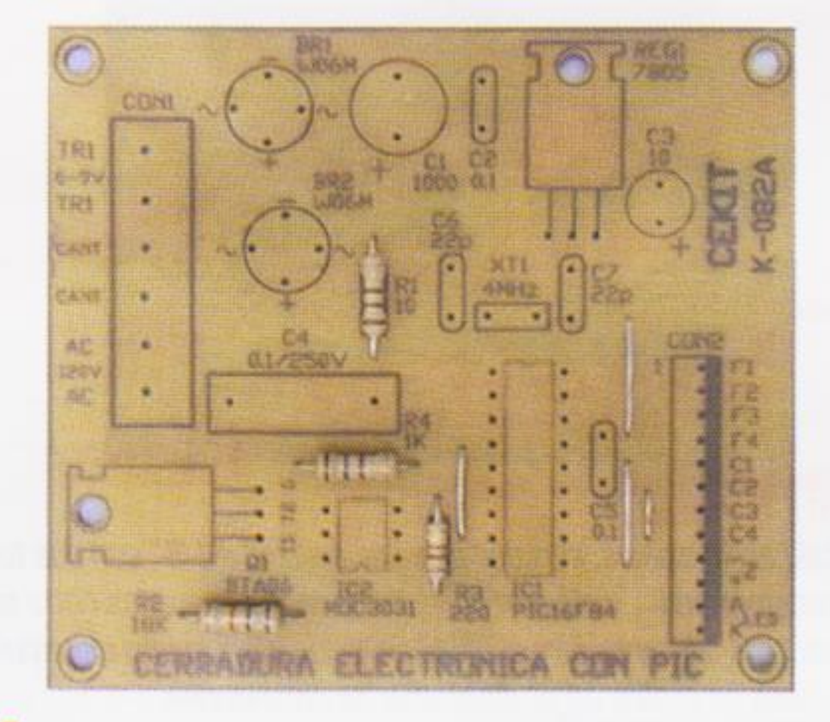
Figura 25.17



Paso 4

En este paso se muestra el aspecto final de la tarjeta del zumbador luego de conectar el cable ribbon de entrada, instalar y pegar el zumbador piezoeléctrico, y haberle colocado su circuito integrado.

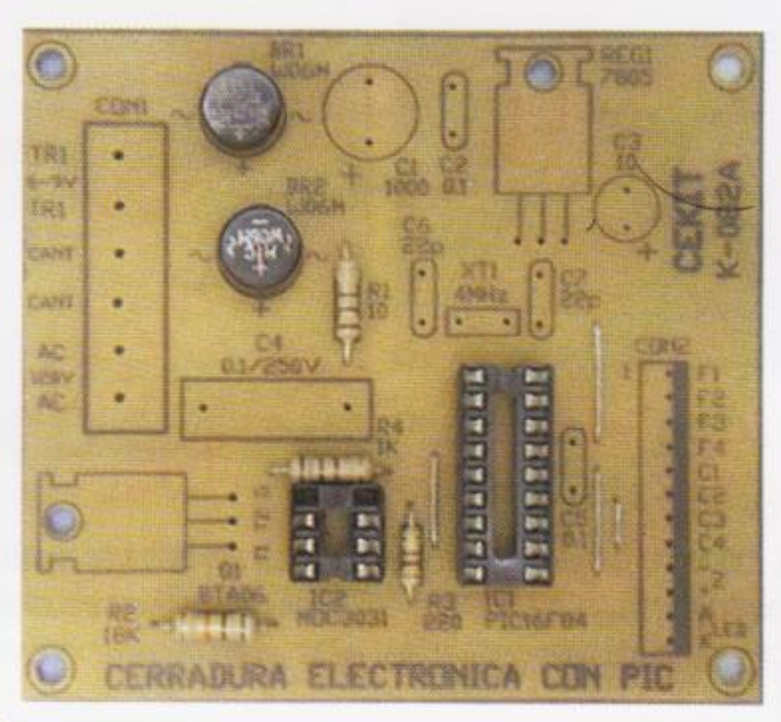
Figura 25.18



Paso 5

Ahora se instalarán todos los componentes de la tarjeta principal de la cerradura electrónica: en este primer paso, instale y suelde los puentes de alambre, luego la resistencia de 1/4 de vatio, y también las tres resistencias de 1/2 vatio.

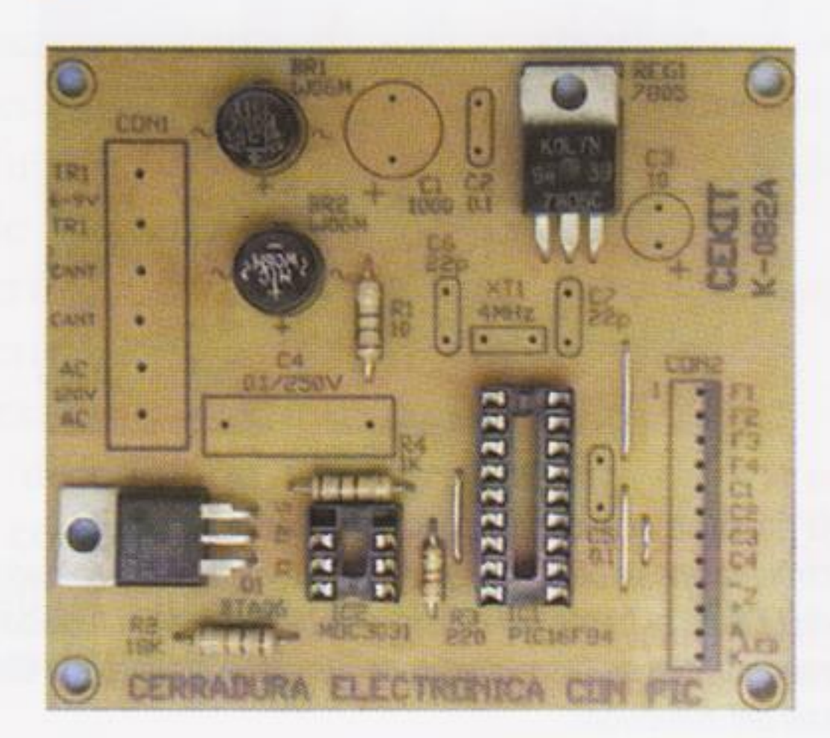
Figura 25.19



Paso 6

Instale y suelde las bases para los circuitos integrados IC1 e IC2, y los dos puentes rectificadores circulares de 1A, BR1 y BR2.

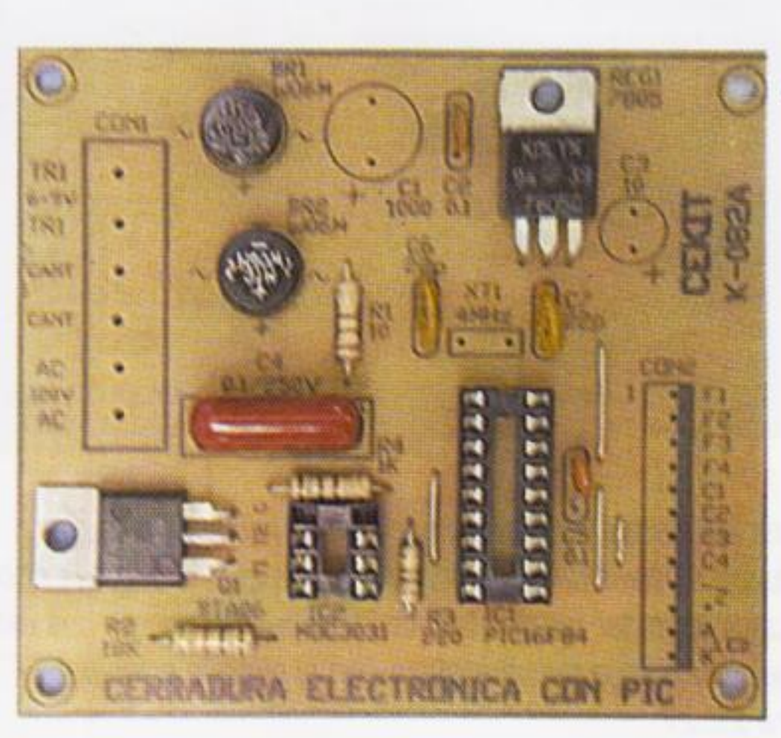
Figura 25.20



Paso 7

Instale y suelde el circuito integrado regulador de voltaje de +5V (REG1), y el TRIAC (Q1) para manejar la cantonera.

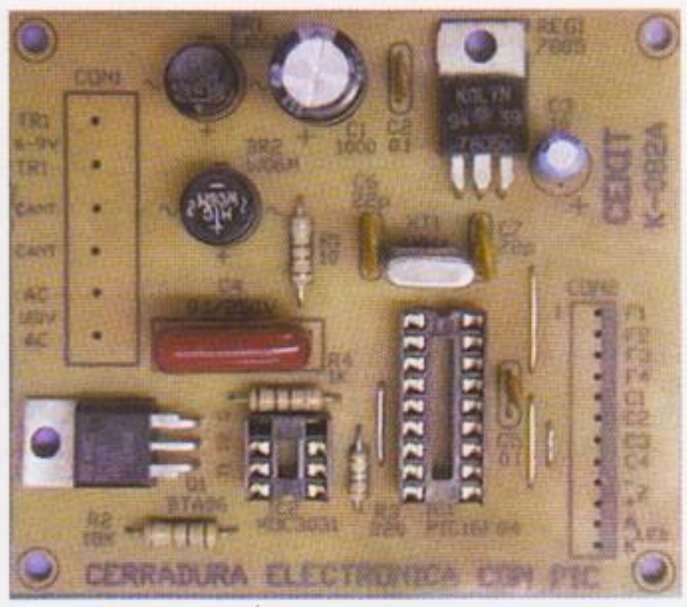
Figura 25.21



Paso 8

Instale y suelde los cuatro condensadores cerámicos de 50V, y también el condensador cerámico de 0,1µF y 250V.

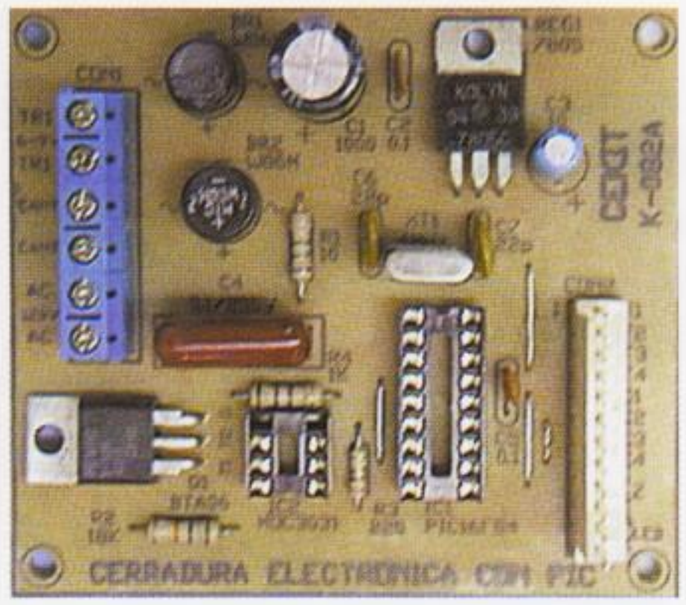
Figura 25.22



Paso 9

Instale y suelde los condensadores electrolíticos C1 y C3, y el cristal de cuarzo de 4MHz del microcontrolador.

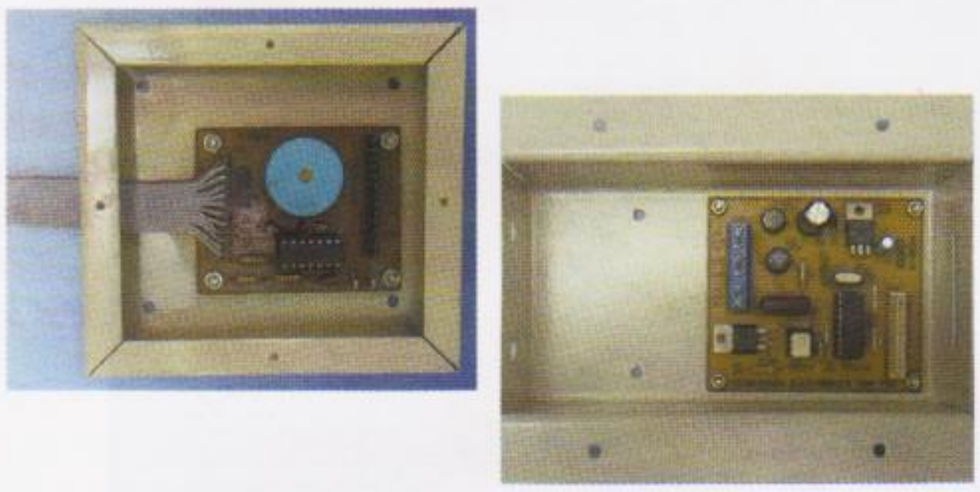
Figura 25.23



Paso 10

Instale y suelde los conectores de tornillo para la entrada de alimentación y la salida de la cantonera, y el conector del teclado y de la tarjeta del circuito zumbador.

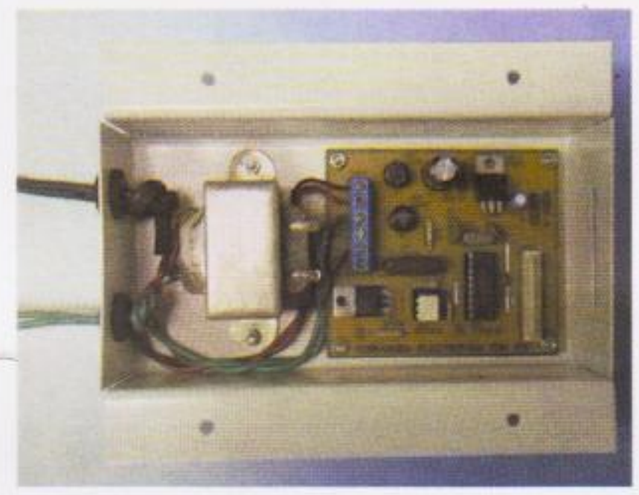
Figura 25.24



Paso 11

Ahora, luego de haber ensamblado las dos tarjetas use cuatro tornillos con tuercas de 1/2" x 1/8" y los respectivos separadores plásticos, para instalar cada una e ellas en su correspondiente chasis.

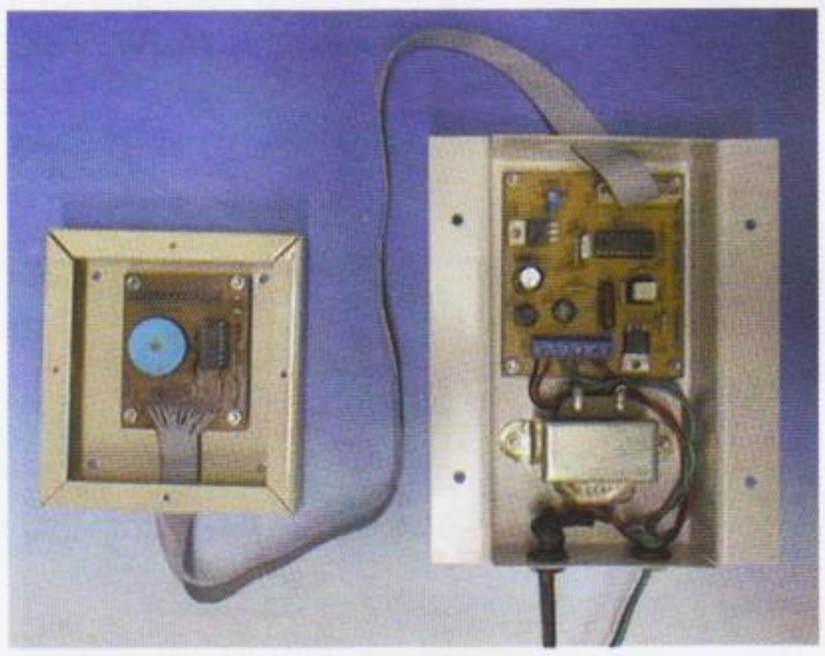
Figura 25.25



Paso 12

Instale en el chasis, con dos tornillos de 1/4" x 1/8" con sus tuercas el transformador , y efectúe las conecciones necesarias entre la entrada de alimentación, el transformador, y el circuito electrónico; tal como se indica en el diagrama de conexiones.

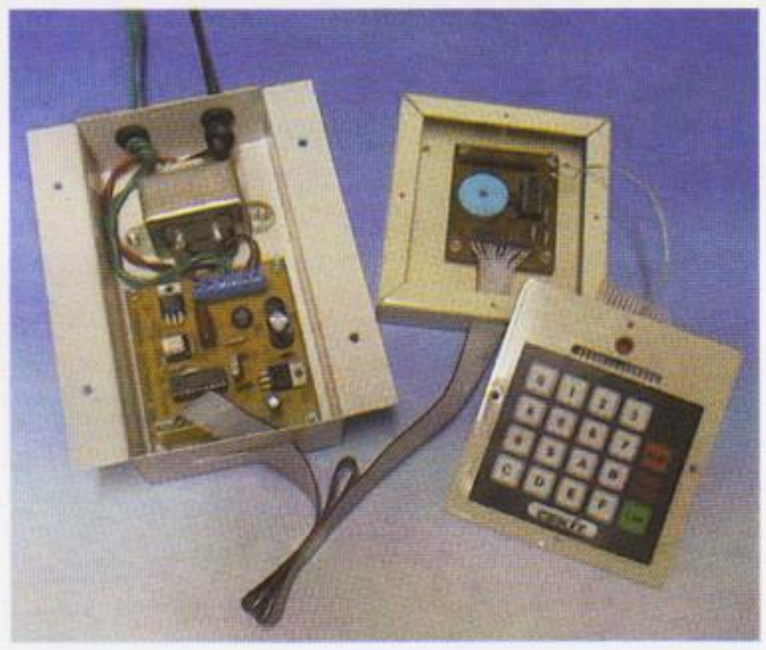
Figura 25.26



Paso 13

Una vez instalados los dos circuitos en su respectivo chasis, introduzca a través de las ranuras de cada chasis el cable ribbon que sirve para conectar las dos tarjetas, y pegue en uno de sus extremos el conector en línea hembra que va en el circuito principal.

Figura 25.27



Paso 14

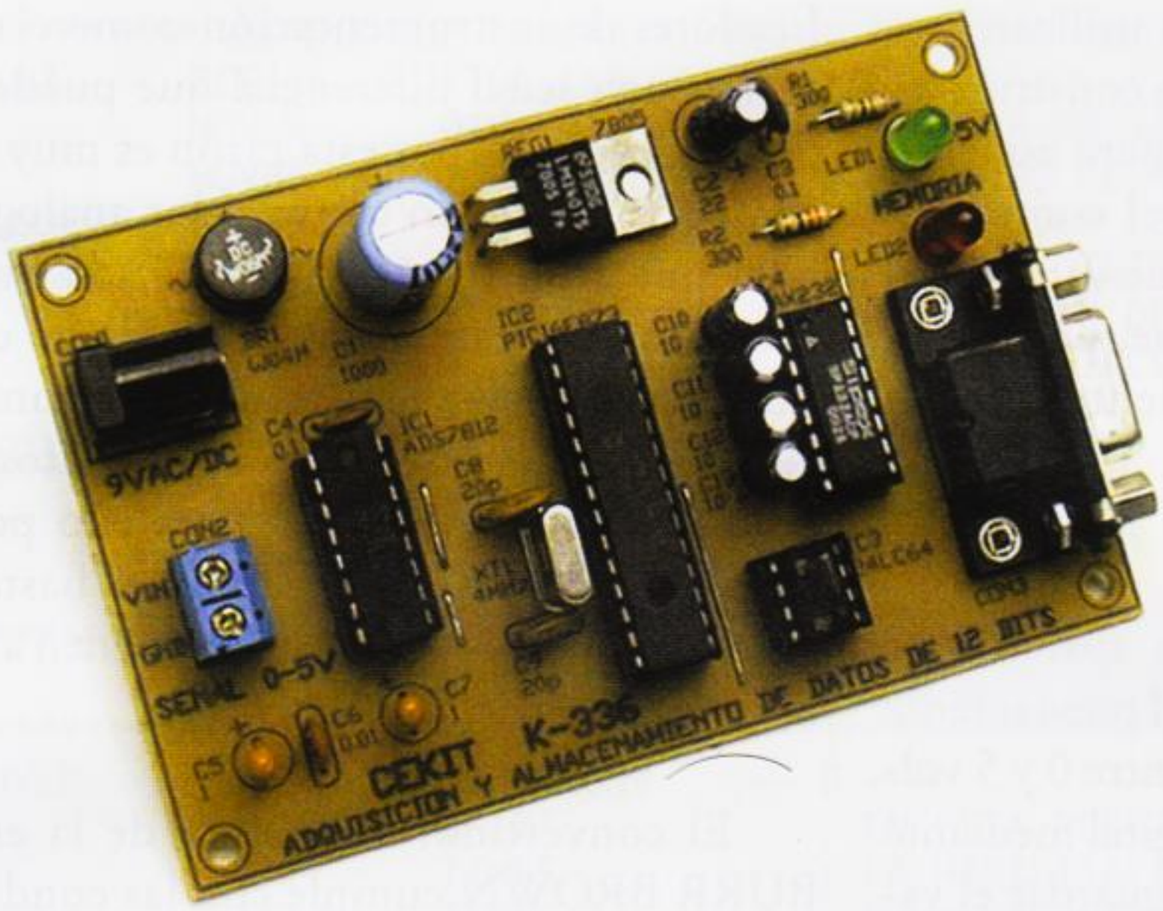
En la figura se muestra el aspecto final de la cerradura electrónica ensamblada, luego de haber instalado y soldado el diodo LED en el frente del chasis, y haber pegado el teclado de membrana, con su respectivo conector, de la tapa frontal del chasis del zumbador.

Figura 25.28

Paso 15. Pruebas finales

Este sistema viene con una configuración inicial para la clave de acceso, verifíquela y compruebe que el circuito del zumbador ejecuta normalmente su trabajo. Luego inserte sus propias claves personales y obtenga un método de seguridad absolutamente confiable y moderno.

PROYECTO 26



REGISTRADOR DE DATOS DE 12 BITS

Existen muchos casos, especialmente en la industria, donde se requiere que los sistemas de instrumentación y/o de medición tengan una resolución de al menos 12 *bits*. Con este proyecto logramos dicho objetivo, utilizando circuitos integrados de reciente incorporación al mercado y con características de funcionamiento muy especiales.

Características

- Adquisición de datos en 12 *bits*
- Almacenamiento de los datos adquiridos dentro de una memoria no volátil
- Transmisión de los datos almacenados por medio de una interfaz RS-232
- Microcontrolador PIC16F873
- Velocidad de transmisión de 9.600 bps

La instrumentación es una de las áreas de mayor interés para quienes trabajamos en electrónica; en ella se pueden desarrollar gran cantidad de proyectos y experimentos que involucran diferentes procesos, dentro de los cuales se puede contar la conversión análogo/digital y la comunicación serial, como algunos de los más frecuentes. En este proyecto utilizaremos justamente estas dos técnicas para construir un equipo que será de gran ayuda para aquellos que deben monitorear o vigilar el comportamiento de variables analógicas que varían lentamente a través del tiempo y que, por su naturaleza, requieren de un tratamiento muy especializado.

Funcionamiento

La idea principal es construir un aparato que permita leer el estado de una señal previamente acondicionada que puede variar entre 0 y 5 voltios, llevar este dato a un valor digital mediante un convertidor A/D de 12 *bits* y guardar el valor de los datos tomados en una memoria no volátil (del tipo EEPROM serial). Posteriormente, los datos de la memoria pueden ser transferidos a una computadora para su análisis y demás procesos requeridos. En la **figura 26.1** se muestra el diagrama de bloques del sistema.

Descripción del circuito

En la **figura 26.2** se muestra el diagrama esquemático del circuito. A continuación describiremos cada uno de los bloques que lo componen. Es importante aclarar que se asume que las señales que se quieren medir han sido acondicionadas previamente por medio de un amplificador de instrumentación.

Convertidor A/D de 12 *bits*. Muchos amplificadores de instrumentación comerciales entregan una señal diferencial que puede variar entre ±5 voltios, por esta razón es muy práctico que se utilice un convertidor análogo a digital que reciba señales de entrada bipolares (positivas y negativas); además, dicho convertidor debe poseer una interfaz de comunicación serial para que entregue los datos al microcontrolador utilizando para ello pocas líneas. Estas condiciones reducen bastante el espectro de posibilidades que se tienen a la hora de escoger dicho dispositivo.

El convertidor ADS7812 de la empresa BURR BROWN cumple con las condiciones especificadas, y aún más, presenta unas características especiales que lo hacen sobresalir entre los demás. A continuación nombramos algunas de ellas:

- Resolución de 12 *bits*
- Múltiples rangos de entrada, incluyendo señales bipolares
- Salida de datos del tipo serial

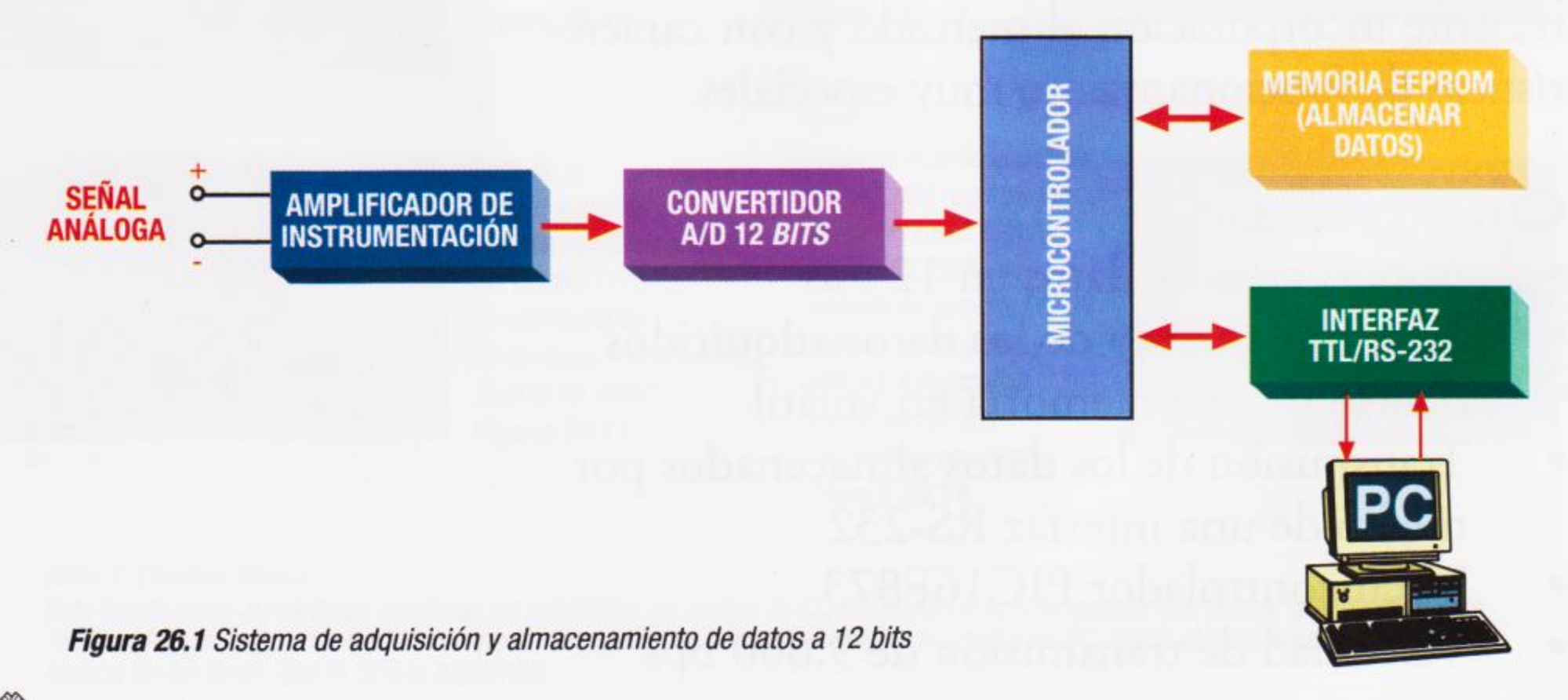


Figura 26.1 *Sistema de adquisición y almacenamiento de datos a 12 bits*

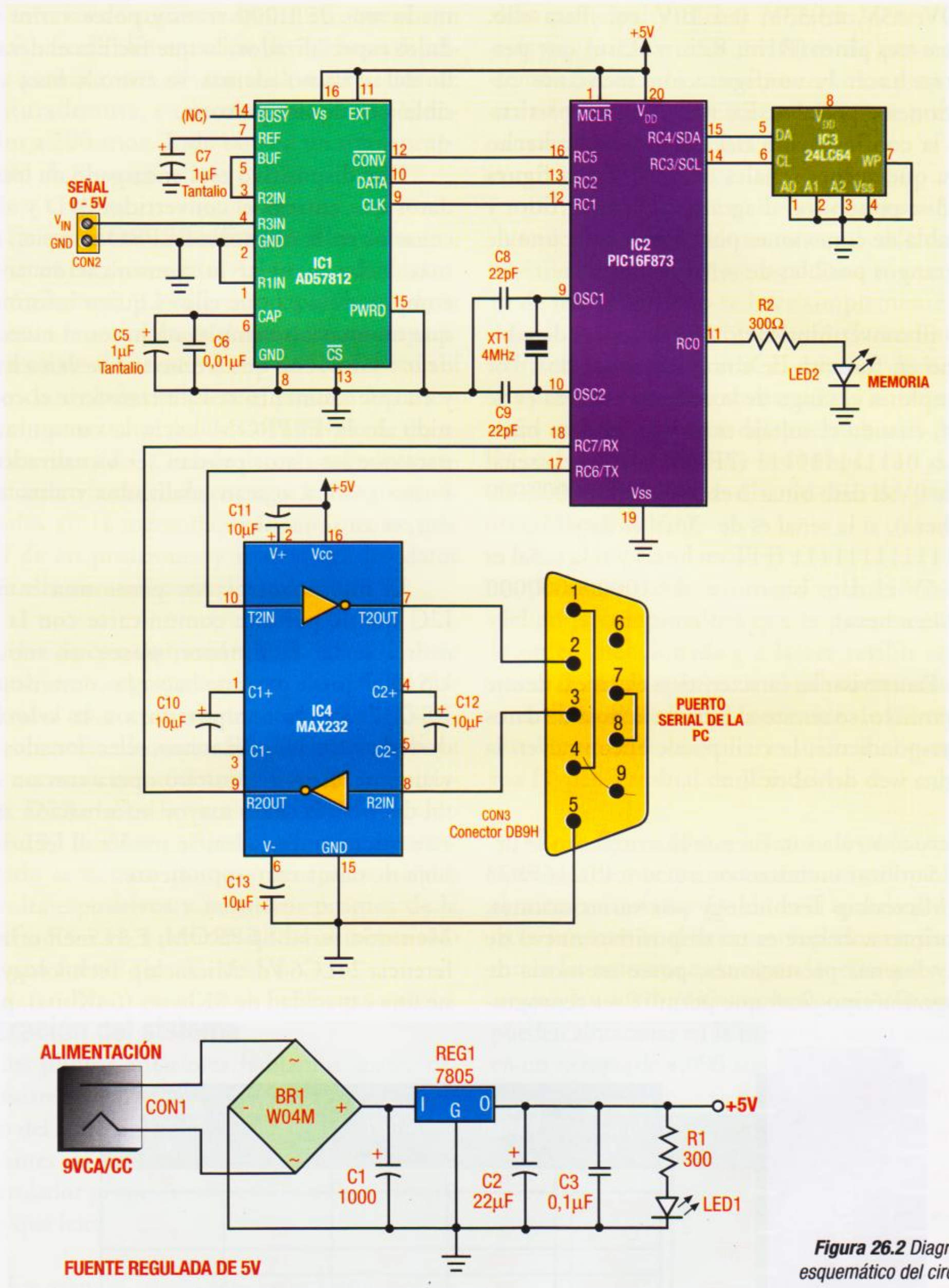


Figura 26.2 *Diagrama esquemático del circuito*

- Opera con una fuente sencilla de +5V
- Tiempo de conversión de 20µs máximo
- Referencia de conversión interna o externa
- Encapsulado tipo DIP de 16 pines

Este dispositivo puede ser configurado para que acepte señales de entrada de diferentes rangos, por ejemplo: de

±10V, ±5V, ±3,33V, 0 a 10V, etc. Para ello, posee tres pines (R1in, R2in y R3in) que permiten hacer la configuración mediante conexiones especiales. En nuestro caso particular, la configuración del circuito se ha hecho para que acepte señales de ±5V. En la **figura 26.3** se presenta el diagrama del convertidor y la tabla de conexiones para lograr cada uno de los rangos posibles de señal de entrada.

El convertidor ADS7812 entrega el dato binario en formato de complemento a dos. Por ejemplo: si el rango de la señal de entrada es de ±5V, cuando el voltaje es de +5V el dato binario es 011111111111 (7FF en hexa), si la señal es de 0V el dato binario es 000000000000 (000 en hexa), si la señal es de -3mV el dato binario es 111111111111 (FFF en hexa) y si la señal es de -5V el dato binario es de 100000000000 (800 en hexa).

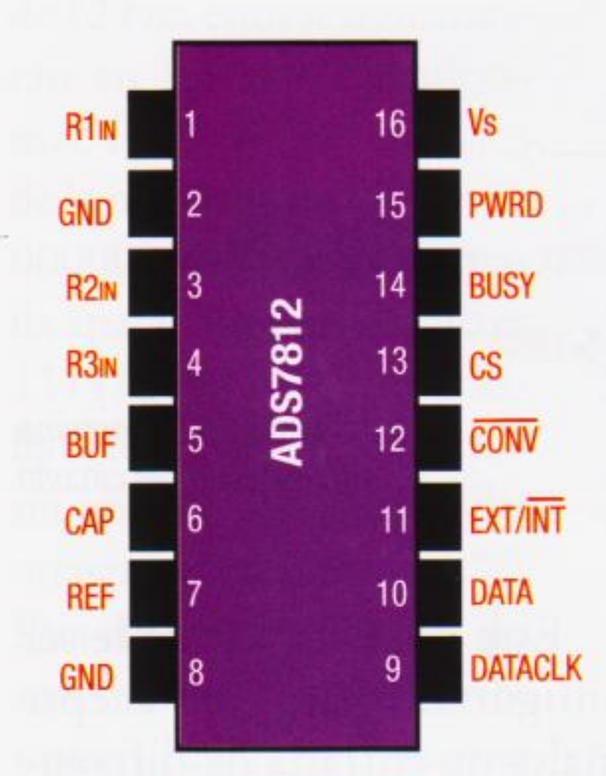
Para revisar las características eléctricas de este dispositivo, se remite al lector a la hoja de datos correspondiente, la cual puede encontrar en la página web del fabricante.

Microcontrolador. En este instrumento se decidió utilizar un microcontrolador PIC16F873 de Microchip Technology por varias razones: la primera, es que es un dispositivo nuevo de muy buenas prestaciones, posee memoria de programa tipo *flash* que permite ser reprogramada más de 1.000 veces y posee varios módulos especializados, lo que facilita el desarrollo del *software*; además, su costo lo hace accesible a cualquier persona.

Este dispositivo está encargado de leer los datos que entrega el convertidor A/D y almacenarlos en la memoria EEPROM serial, además, debe manejar la comunicación con la computadora ya que ella es quien informa en que momento se empieza a hacer el muestreo de los datos, con que frecuencia se van a hacer, y en que momento se va a transferir el contenido de la EEPROM hacia la computadora para que los datos puedan ser visualizados en forma gráfica, o sean analizados y almacenados, si así se requiere.

El microcontrolador posee una interfaz I2C que le permite comunicarse con la memoria serial fácilmente, posee un módulo USART que le permite hacer la comunicación RS-232 con la computadora a la velocidad deseada y con los parámetros seleccionados previamente. Este dispositivo opera con un cristal de 4 MHz. Para mayor información sobre este microcontrolador, se remite al lector a la hoja de datos correspondiente.

Memoria serial EEPROM. Esta memoria, referencia 24LC64 de Microchip Technology, tiene una capacidad de 8Kbytes (64Kbits), posee



RANGO DE ENTRADA (V)	CONECTAR A R1IN	CONECTAR A R2IN	CONECTAR A R3IN	IMPEDANCIA DE ENTRADA (KΩ)
±10V	VIN	BUF	GND	45.7
0,3125V a 2,8125V	VIN	VIN	VIN	>10,000
±5V	GND	BUF	VIN	26,7
0V a 10V	BUF	GND	VIN	26,7
0V a 4V	BUF	VIN	GND	21,3
±3,33V	VIN	GND	VIN	21,3
0,5V a 4,5	GND	VIN	GND	21,3

Figura 26.3 *Convertidor A/D de 12 bits con interfaz serial y rango de señal de entrada configurable. ADS7812.*

una interfaz de comunicación I2C, opera con una fuente de alimentación de 2,5 hasta 5 voltios, permite más de un millón de ciclos de escritura/lectura, y una retención de datos superior a 200 años. Todo esto en un solo encapsulado de 8 pines.

En esta memoria se almacenan los datos que provienen del convertidor A/D. Como cada conversión entrega 12 *bits*, se ocuparán dos *bytes* por cada muestra, lo cual nos deja una capacidad total de almacenamiento de 4K. Cuando el microcontrolador empiece el proceso de muestreo almacenará un dato cada intervalo de tiempo determinado. Cuando la computadora envíe al micro la orden de entregar los datos almacenados en la memoria, éste hará un barrido total de las posiciones y transmitirá los datos hacia ella uno tras otro.

Interfaz de comunicación TTL/RS-232. Este bloque se encarga de convertir las señales que maneja el microcontrolador en niveles TTL (0-5V), a niveles adecuados RS-232 para que haya una correcta comunicación con la computadora. Dicha labor la hace el circuito integrado MAX232 de Maxim Semiconductor. Este integrado se destaca porque maneja los niveles de voltaje positivos y negativos propios de la interfaz RS-232, a pesar de que se alimenta sólo con una fuente sencilla de 5V.

Operación del sistema

En los puntos anteriores hablamos acerca del *hardware* del proyecto; ahora, vamos a hablar un poco del *software* que es quizás lo más interesante. Antes de referirnos al programa del microcontrolador propiamente, vamos a describir que es lo que hace.

En esencia, el equipo debe tomar muestras de la señal de entrada, convertirlas a un equivalente digital de 12 *bits* y almacenar dichos datos en una memoria no volátil. Las muestras deben ser tomadas a intervalos regulares de tiempo, establecidos por la computadora principal y dichas muestras deben además ser tratadas con un filtro de promedio dentro del mismo microcontrolador para evitar así errores de lectura causados por interferencias transitorias en la señal.

El proceso de toma de muestras se inicia cuando la computadora envía al microcontrolador una orden que especifica el intervalo de muestreo; esto se hace simplemente enviando un número entre 1 y 9 (la computadora envía el código ASCII de estos dígitos al ser pulsada la tecla respectiva). La comunicación se hace a través del puerto serial de la PC a una velocidad de 9.600bps, con datos de 8 *bits*, sin paridad y un bit de parada (9.600bps, 8, n,1).

El usuario debe crear una aplicación especial en la computadora que le permita enviar la orden mencionada y a la vez recibir el paquete de datos que envía el microcontrolador cuando se desea leer el contenido de la memoria. Una opción sencilla es utilizar el programa HyperTerminal de Windows.

En la **figura 26.4** se muestra la tabla con los intervalos de muestreo y demás funciones que hace el microcontrolador. De ahí, si el usuario pulsa la tecla número 1, el microcontrolador empieza a tomar las muestras cada segundo, esto quiere decir que las 4.096 (4K) conversiones A/D que se pueden almacenar en la memoria, serán tomadas en un tiempo de 4.096 segundos.

Tecla pulsada	Función
1	Intervalo 1 s
2	Intervalo 2 s
3	Intervalo 4 s
4	Intervalo 8 s.
5	Intervalo 16 s
6	Intervalo 32 s
7	Intervalo 64 s
8	Intervalo 128 s
9	Intervalo 255 s
Enter	Lectura de la memoria

Figura 26.4 *Funciones del equipo (tiempos de muestreo)*

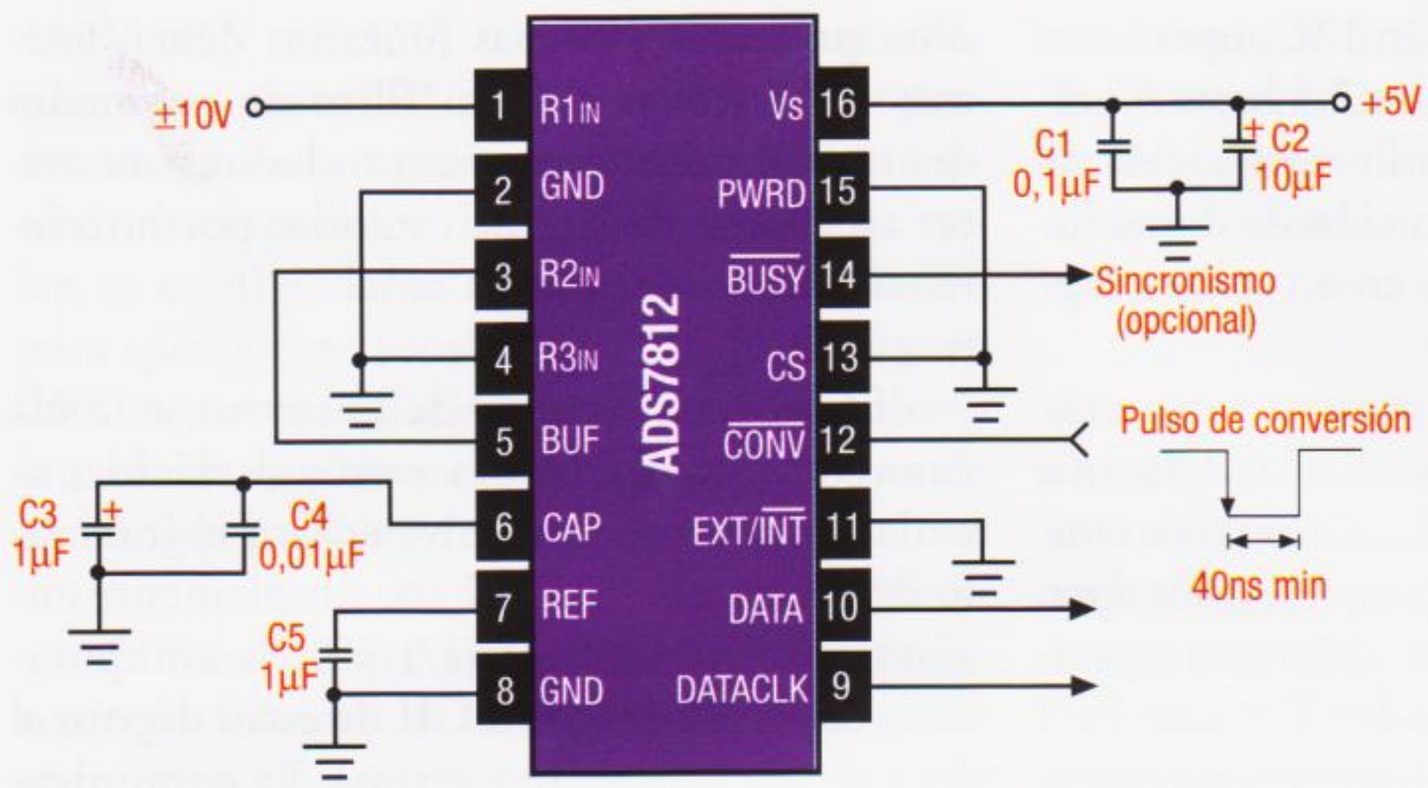


Figura 26.5 *Circuito para el manejo del conversor ADS7812*

Así sucesivamente se puede calcular el tiempo para cada intervalo seleccionado. El LED conectado al microcontrolador titila cada vez que toma una muestra y permanece encendido cuando se llena el contenido de la memoria EEPROM.

Una vez que la memoria está llena, se debe vaciar su contenido en la computadora, para ello el usuario debe pulsar la tecla Enter (con esto se envía al microcontrolador el código ASCII de la tecla que es 0D en hexadecimal). El microcontrolador, de inmediato, comienza a leer las posiciones de memoria y a transmitir hacia la PC los datos correspondientes.

Como se trata de datos de 12 *bits*, éstos se transmitirán en formato hexadecimal. Por ejemplo, si el dato de la memoria en binario es 000000000000, en pantalla aparece 000; si el dato es 111111111111, en pantalla aparece FFF. Cada muestra aparece separada de las otras por el carácter coma ",". Para obtener el programa del microcontrolador visite nuestra página en la internet **www.cekit.com.co.**

Dentro de las aplicaciones prácticas que puede tener este dispositivo se encuentran los monitores de silos o ambientes con variables que deben ser controladas, tales como temperatura, presión, humedad, etc. Otra aplicación es el seguimiento de la temperatura en invernaderos, donde los cambios del clima durante la noche pueden afectar considerablemente el desarrollo de los cultivos.

Operación del ADS7812

En la **figura 26.5** se muestra un circuito básico para trabajar con el ADS7812. Para empezar una conversión y la transmisión serial de los resultados de la conversión previa, se requiere de un flanco de bajada en el pin CONV.

El pin BUSY se pondrá en estado lógico bajo para indicar que la conversión se ha iniciado y permanecerá así hasta que finalice. Durante la conversión, los resultados de la conversión anterior se transmitirán por el pin DATA mientras el pin DATACLK proporciona el reloj de sincronización para los datos seriales.

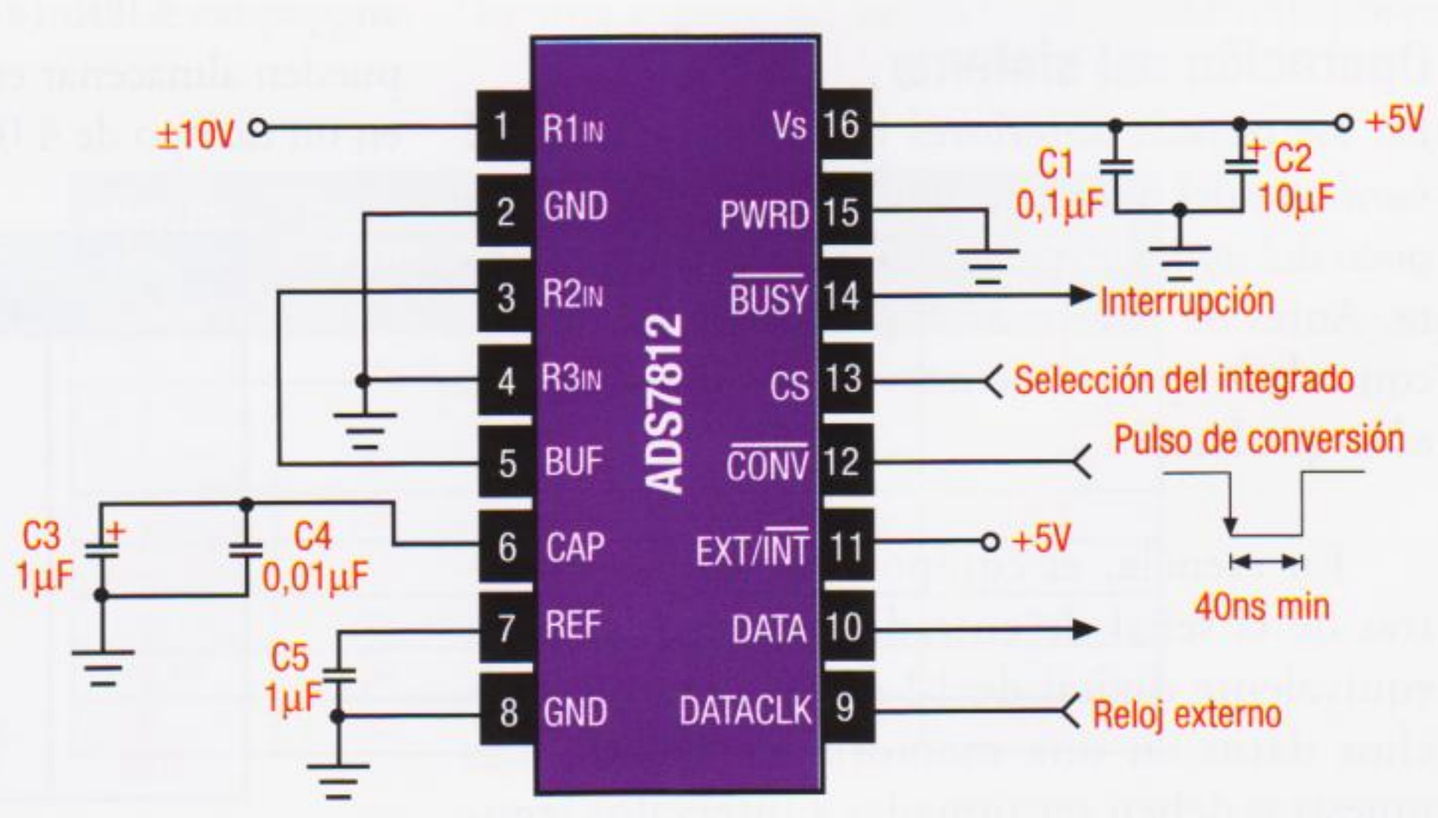


Figura 26.6 *Circuito para el manejo del conversor ADS7812 con señal externa de reloj*

Otro modo de trabajo del conversor es con una señal externa en el pin DATACLK, **figura 26.6**. En este caso el resultado se almacena en un registro de desplazamiento que se basa en la señal de reloj generada por un dispositivo externo, en este caso el microcontrolador. En la **figura 26.7** se aprecia una parte del código fuente en donde se generan las señales para acceder a los datos del convertidor, se puede apreciar en la parte A la generación de la señal de DATACLK, y en la parte B, la generación de la señal para el pin CONV, de acuerdo a lo explicado en las figuras anteriores.

```
;********** RUTINA QUE GENERA EL PULSO DE RELOJ PARA LEER EL ADS7812************
RELOJ     BCF       PTOC,CLK    ; -- A ---
          MOVLW     01H                 ; EN ESTA PARTE DEL PROGRAMA SE GENERA
          MOVWF     TIMES               ; EL PULSO DE RELOJ (PIN CLK) NECESARIO
RELOJ2    DECFSZ    TIMES,1             ; PARA PODER LEER EL CONVERTIDOR
          GOTO      RELOJ2              ; ANÁLOGO DIGITAL ADS7812.
          BSF       PTOC,CLK            ; EL PIN CLK DEL PUERTO C VA CONECTADO
          MOVLW     01H                 ; AL PIN DATACLK DEL INTEGRADO.
          MOVWF     TIMES
RELOJ3    DECFSZ    TIMES,1
          GOTO      RELOJ3
          RETURN

;****************RUTINA DE CONVERSIÓN A/D DEVUELVE 2 BYTES*********************
;*              8 BITS BAJOS EN ADRESL Y 4 BITS ALTOS EN ADRESH              *
;*****************************************************************************
CONVER    BCF       PTOC,CONV   ;-- B ---
          MOVLW     D'10'               ; EN ESTA PORCIÓN DEL PROGRAMA SE ENVÍA
          MOVWF     TIMES               ; LA HABILITACIÓN PARA EL PIN
CONVE2    DECFSZ    TIMES,1             ; CONV DEL CIRCUITO ADS7812.
          GOTO      CONVE2
          CLRF      ADRESH              ;LIMPIAR REGISTRO DE RECEPCIÓN
          CLRF      ADRESL
          MOVLW     04H                 ;INICIA CONTADOR PARA LEER 2 BITS DE MAYOR
                                        ;PESO
          MOVWF     CONTA
          BCF       PTOC,CONV           ;BAJA CHIP SELECT PARA EMPEZAR
          NOP
OTROBIT   CALL      RELOJ               ;LLAMA PULSO DE RELOJ
          BSF       STATUS,C            ;LOS DATOS SE VAN ALMACENANDO EN FORMATO
          BTFSS     PTOC,IN             ;SERIAL. EL BIT QUE VA LLEGANDO SE VA
          BCF       STATUS,C            ;ROTANDO SOBRE LOS REGISTROS ADRESH Y
          RLF       ADRESH,1            ; ADRESL.
          DECFSZ    CONTA,1
          GOTO      OTROBIT
          MOVLW     08H                 ;INICIA CONTADOR PARA LEER 8 BITS DE MENOR
          MOVWF     CONTA               ; PESO
          BCF       PTOC,CONV           ;BAJA CHIP SELECT PARA EMPEZAR
          NOP
OTROBIS   CALL      RELOJ
          BSF       STATUS,C
          BTFSS     PTOC,IN
          BCF       STATUS,C
          RLF       ADRESL,1
          DECFSZ    CONTA,1
          GOTO      OTROBIS
          BSF       PTOC,CONV
          BCF       PTOC,CLK
          NOP
          RETURN
```

Figura 26.7 *Porción del código fuente donde se generan las señales para el convertidor*

Ensamblaje del proyecto

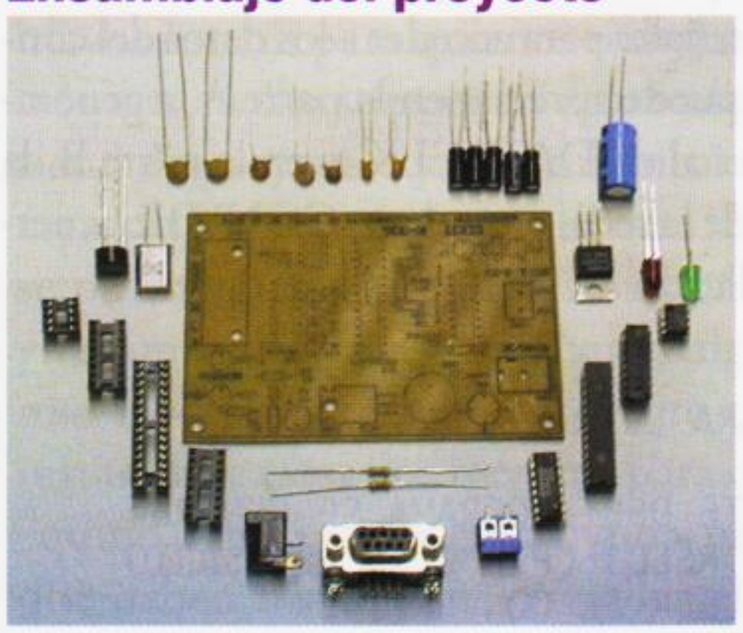
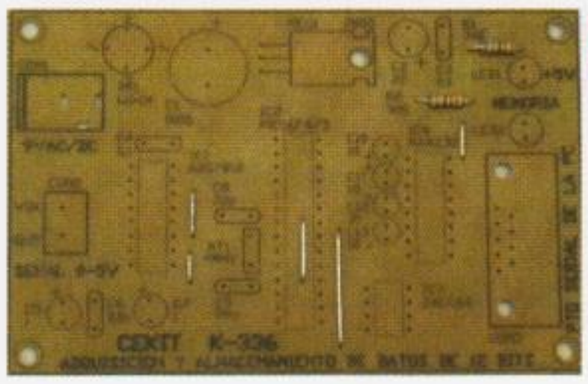


Figura 26.8 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el circuito del registrador de datos de 12 bits CEKIT K-337.*

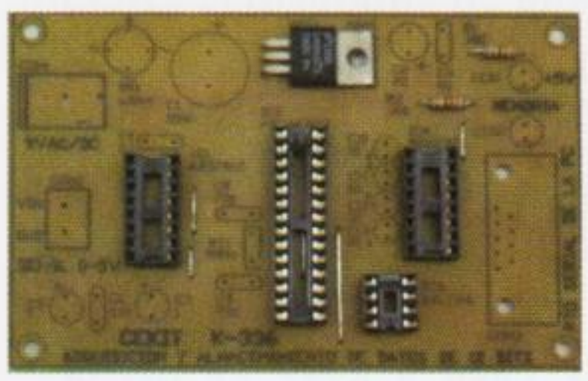
Por medio de este circuito se pueden adquirir datos provenientes de sensores o transductores con un alto grado de resolución, de 12 bits. Con este proyecto se facilita la toma y almacenamiento de datos en el laboratorio o en el campo, y el posterior análisis en la computadora. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

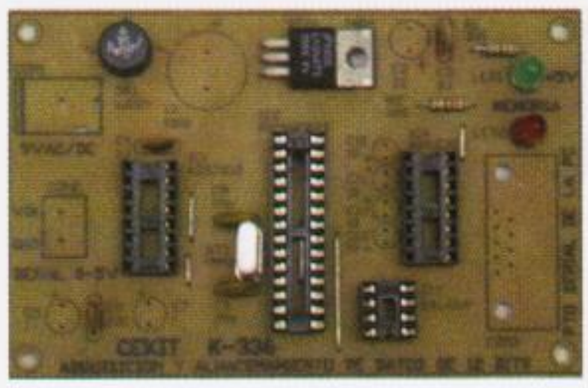
Cant.	Ref.	Descripción
1	IC1	Circuitos integrado ADS7812
1	IC2	Microcontrolador PIC 16F873, programado
1	IC3	Circuito integrado 24LC64
1	IC4	Circuitos integrado MAX232
1	REG1	Regulador fijo de +5V -1A, 7805
1	BR1	Puente rectificador circular de 1A, W04M
1	LED1	Diodo LED de 5mm, verde
1	LED2	Diodo LED de 5mm, rojo
1	C1	Condensador electrolítico de 1000uF - 25V
1	C2	Condensador electrolítico de 22uF - 16V
2	C3 y C4	Condensadores cerámicos de 0.1uF - 50V
2	C5 y C7	Condensadores de tantalio de 1uF - 35V
1	C6	Condensador cerámico de 0.01uF - 50V
2	C8 y C9	Condensadores cerámicos de 22pF - 50V
4	C10 a C13	Condensadores electrolíticos de 10uF - 16V
2	R1 y R2	Resistencias de 300 - 1/4W
1	XT1	Cristal de cuarzo de 4MHz
1	K-337	Circuito impreso CEKIT
1	CON1	Conector AC/DC para alimentación
1	CON2	Conector de tornillo de 2 pines para impreso
1	CON3	Conector DB 9 Hembra para circuito impreso
1		Base de 8 pines para circuito integrado
2		Bases de 16 pines para circuito integrado
1		Bases de 28 pines para circuito integrado



Paso 1
Instale y suelde los puentes de alambre telefónico desnudo, y las dos resistencias de 1/4 de vatio.
Figura 26.9

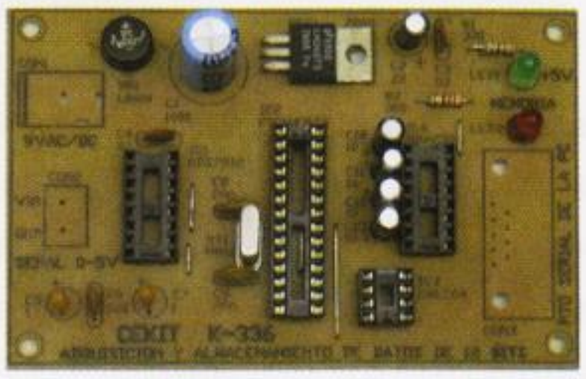


Paso 2
Luego instale y suelde las bases para los circuitos integrados y el regulador de voltaje de +5V.
Figura 26.10

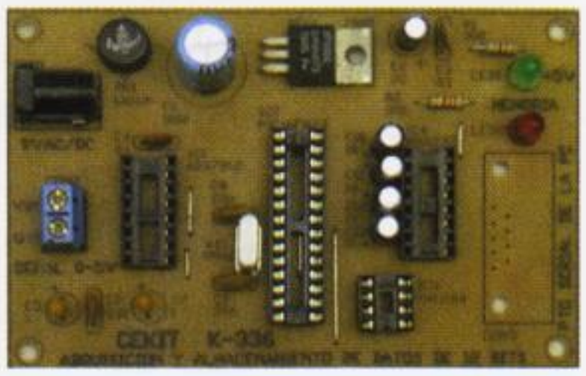


Paso 3
Instale y suelde el puente rectificador BR1, los diodos LED, los condensadores cerámicos, y el cristal de cuarzo de 4MHz.
Figura 26.11

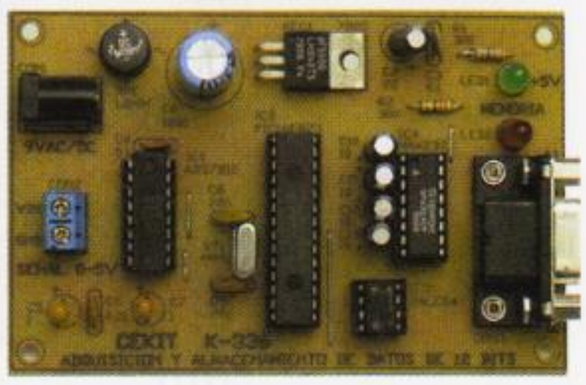
Paso 4
Instale y suelde los dos condensadores de tantalio, y los condensadores electrolíticos.
Figura 26.12



Paso 5
Instale y suelde el conector para la alimentación de la tarjeta (9VCA/CC), y los bornes para la señal de entrada entre 0 y 5V.
Figura 26.13

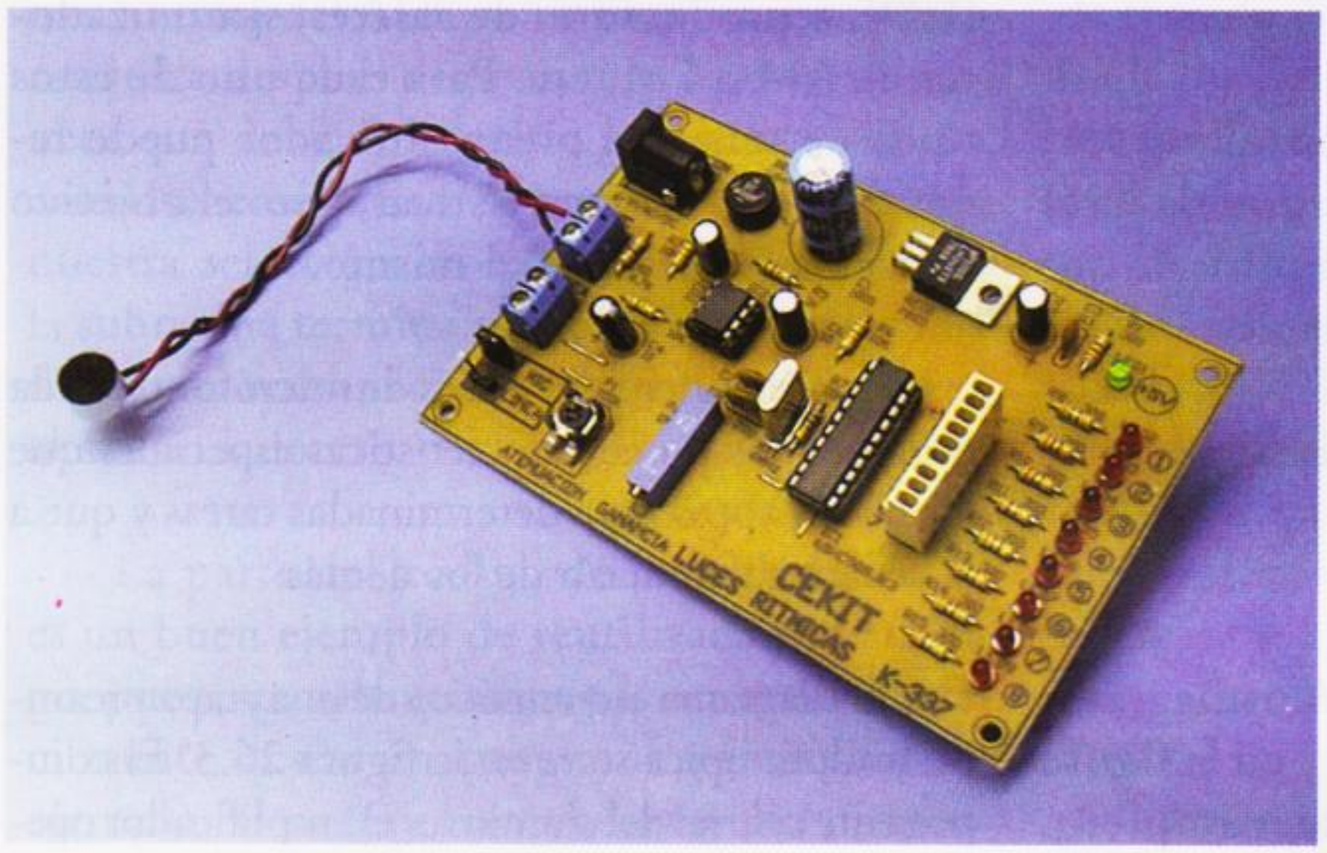


Paso 6
Aspecto final de la tarjeta del circuito registrador de datos de12 bits K-337, luego de haber soldado el conector DB-9H para el puerto serial, y ubicado los circuitos integrados en sus respectivas posiciones.
Figura 26.14



Paso 7. Pruebas finales
Esta tarjeta debe alimentarse mediante un adaptador de voltaje de CC sin importar la polarización que posea, gracias al puente rectificador y al regulador de 5 V. Por medio del puerto serial y un programa de prueba de PC, como el hyperterminal, haga una lectura de los datos que la tarjeta transmite.

PROYECTO 26



LUCES RÍTMICAS

En este proyecto construiremos un circuito que genera secuencialmente unos patrones de luces, y, dependiendo de una entrada de audio, se varía la velocidad de la secuencia visualizada.

Características

- Envío de secuencia de 8 *bits* a través de diodos LED
- La velocidad con que se origina la secuencia varía de acuerdo a una entrada de audio
- Lectura de señal análoga proveniente de un micrófono u otra fuente de sonido
- Microcontrolador Motorola

Funcionamiento

En muchos sitios públicos, como tabernas, espectáculos y discotecas, se tienen dispositivos de luces para ambientar los distintos escenarios. La variedad y creatividad con que estas luces se enciendan y apaguen, combinados con las características del sonido en cada instante del espectáculo pueden crear diversas emociones en el observador, lo que hace a estos sistemas muy atractivos comercialmente.

En este proyecto vamos a construir un sencillo dispositivo que genera un patrón de luces cuya velocidad depende de la intensidad con que se sienta la música. Si el usuario lo desea puede filtrar las señal del sonido y variar la velocidad de las luces, dependiendo de los sonidos bajos, medios y altos.

Descripción del circuito

En la **figura 26.1** observamos un diagrama de bloques general del proyecto y en la **figura 26.2** el diagrama esquemático completo. Como podemos apreciar, existen dos posibles clases de entradas al secuenciador: entrada de micrófono o entrada para otra fuente de sonido. Ambas posibilidades deben ser previamente sometidas a una etapa de preamplificación que adecúa sus débiles señales para que puedan ser leídas desde el conversor A/D interno del microcontrolador.

Etapa de preamplificación

La señal que entrega un micrófono tiene niveles de voltaje y de corriente muy bajos. Por esa razón no se pueden conectar directamente a circuitos amplificadores ni a parlantes. Para solucionar este incoveniente, se usan los circuitos preamplificadores, los cuales pueden aceptar señales de entrada de muy bajo nivel y las amplifican hasta valores adecuados para ser introducidas en un amplificador de potencia. Los preamplificadores también se utilizan con otras fuentes de señal, como tocadiscos, reproductores de discos compactos (CD), reproductores de casetes, sintonizadores de AM o FM, etc. Para cada uno de estos tipos de señal, el preamplificador puede tener diferentes características, pero el objetivo en todos los casos es el mismo.

Existen diferentes clases de micrófonos, cada uno de ellos posee características especiales que los hacen aptos para determinadas tareas y que a la vez los diferencian de los demás.

El diagrama esquemático de una etapa preamplificadora típica se ve en la **figura 26.3**. El componente central del circuito es el amplificador operacional LM358, éste cumple la función de amplificar las débiles señales que entrega el micrófono para llevarlas a niveles adecuados para ser aplicadas a un amplificador de potencia. Para ajustar el nivel de amplificación a un valor adecuado se utiliza el *trimmer* P2, el cual debe ajustarse cada vez que se cambie de micrófono.

Entre los pines 1 y 2 del amplificador operacional (salida y entrada negativa respectivamente) se ha conectado una resistencia ajustable (*trimmer*) de 1MΩ, éste sirve para ajustar el nivel de amplificación al nivel deseado. Normalmente la

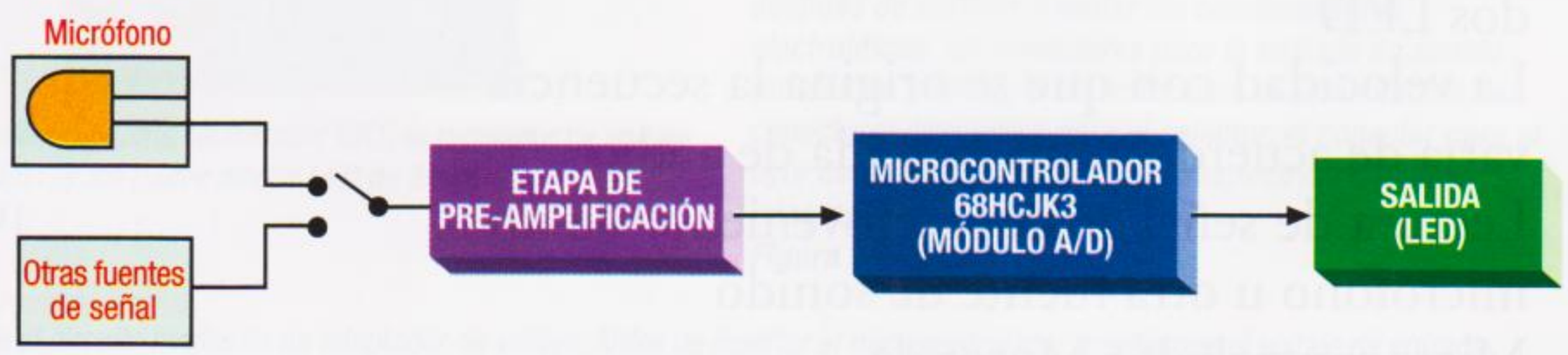


Figura 26.1 *Diagrama de bloques del circuito*

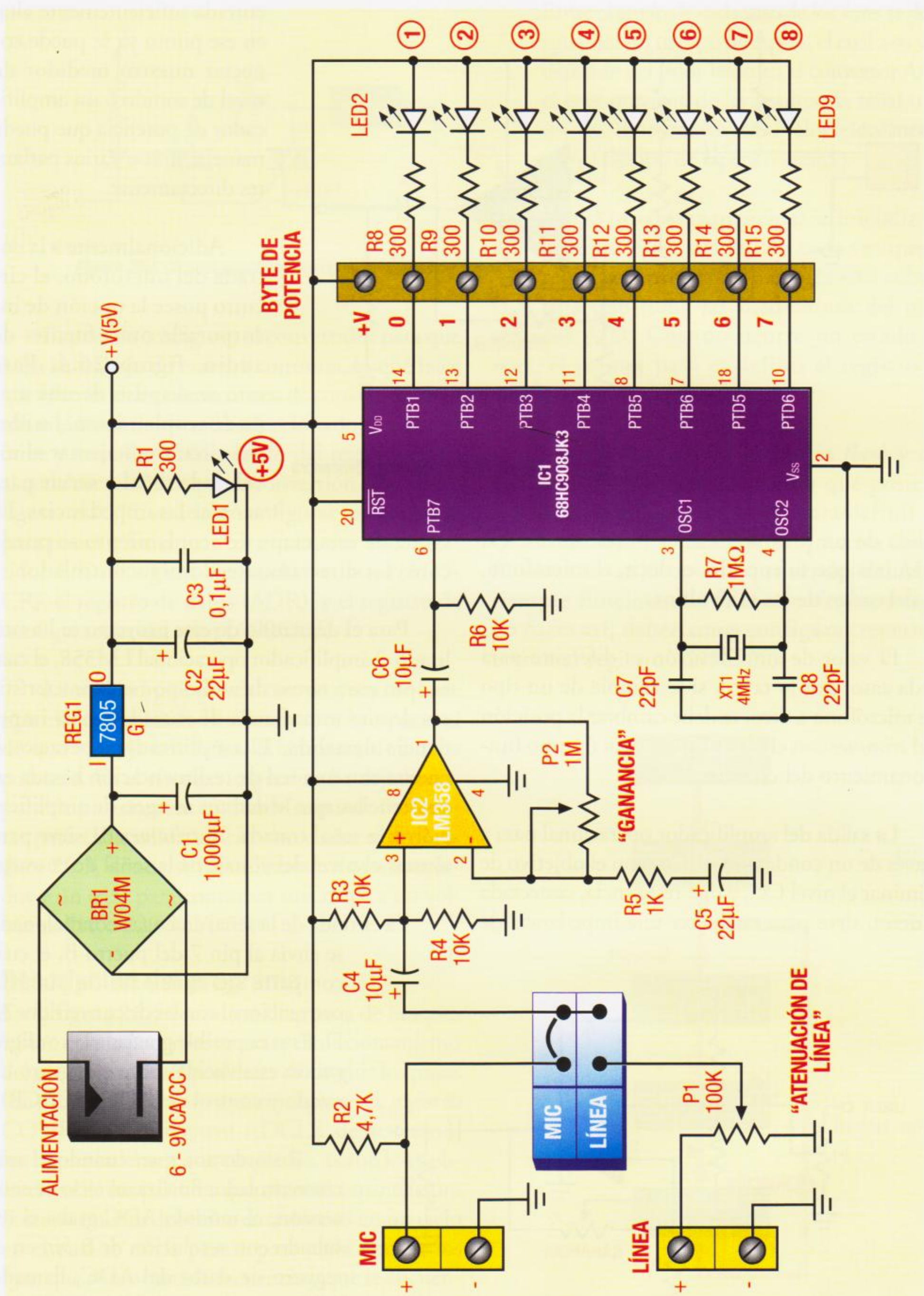


Figura 26.2 Diagrama esquemático del secuenciador de luces

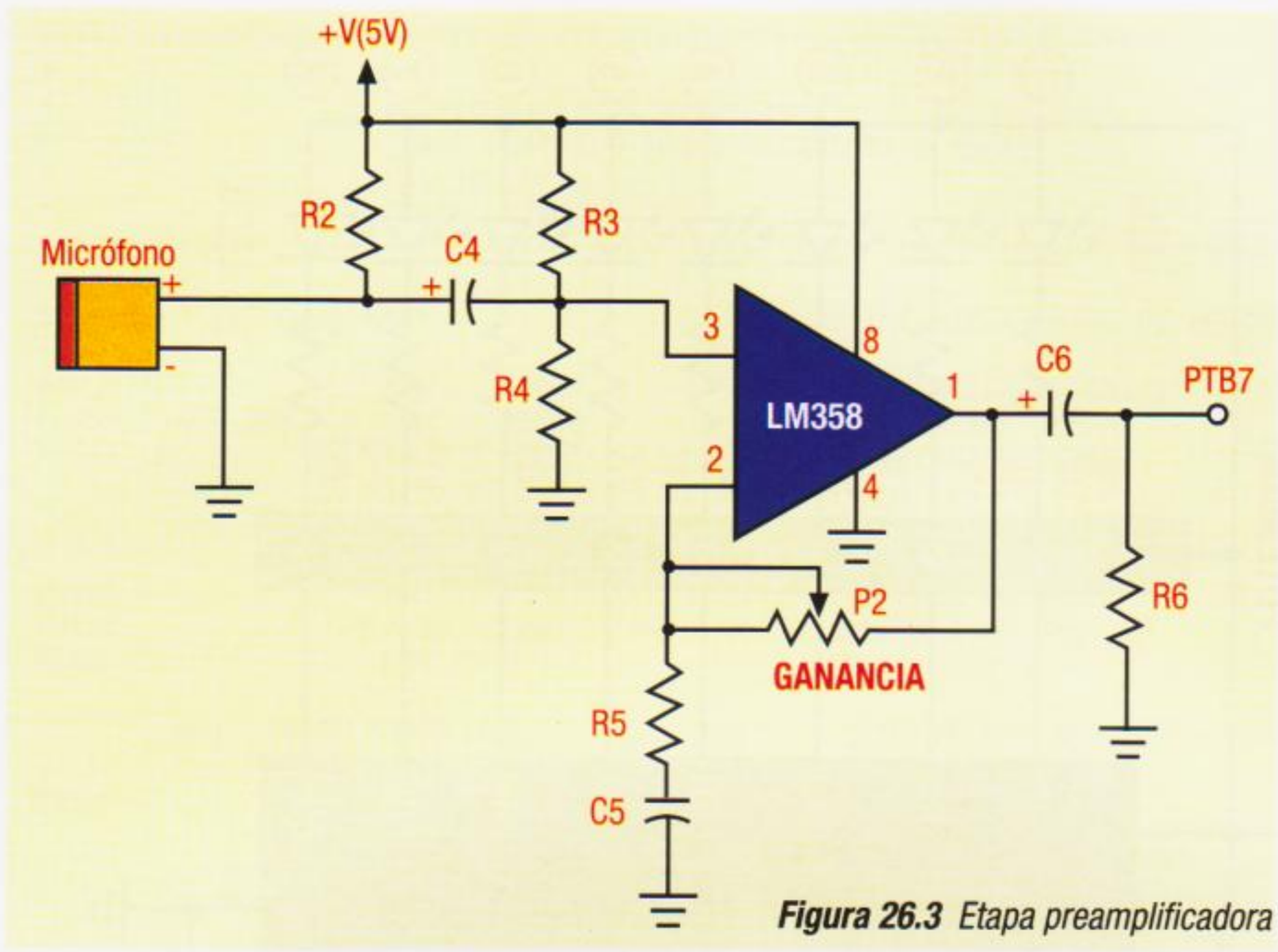


Figura 26.3 *Etapa preamplificadora*

salida de un preamplificador puede ser de 1V, mientras que su entrada, es decir, el micrófono, es del orden de los milivoltios.

El valor de amplificación es diferente para cada caso, por lo tanto, si se cambia de un tipo de micrófono a otro, se debe cambiar la posición del *trimmer* con el fin de obtener un óptimo funcionamiento del circuito.

La salida del amplificador operacional pasa a través de un condensador (C6) con el objetivo de eliminar el nivel CC, y una resistencia, conectada a tierra, sirve para establecer una impedancia de entrada suficientemente alta; en ese punto ya se puede conectar nuestro medidor de nivel de sonido o un amplificador de potencia que pueda manejar uno o varios parlantes directamente.

Adicionalmente a la entrada del micrófono, el circuito posee la opción de incorporarle otras fuentes de audio, **figura 26.4**. Para esto se dispuso de una etapa de acoplamiento. La idea del circuito es ajustar el nivel de la señal y servir para acoplar las impedancias. La salida de esta etapa de acoplamiento se puede conectar directamente al microcontrolador.

Para el desarrollo de este proyecto se ha utilizado el amplificador operacional LM358, el cual es apto para tareas de audio por sus características de alta impedancia de entrada y baja impedancia de salida. El amplificador operacional cuenta con una red de realimentación basada en resistencias, que le dan un margen de amplificación a la señal tratada. El *trimmer* P1 sirve para ajustar el nivel del voltaje de la señal de entrada.

La entrada de la señal análoga acondicionada se envía al pin 7 del puerto B, el cual comparte sus funciones de pin I/O convencional con las de convertidor A/D. Esto es posible gracias a la configuración establecida sobre el registro de estado y control del ADC (ADSCR).

Recordemos que, cuando el microcontrolador finaliza un ciclo de conversión, el módulo ADC graba el resultado con resolución de 8 *bits* en el registro de datos del ADC, llamado ADR; y pone en uno lógico su correspondiente *bit* bandera COCO.

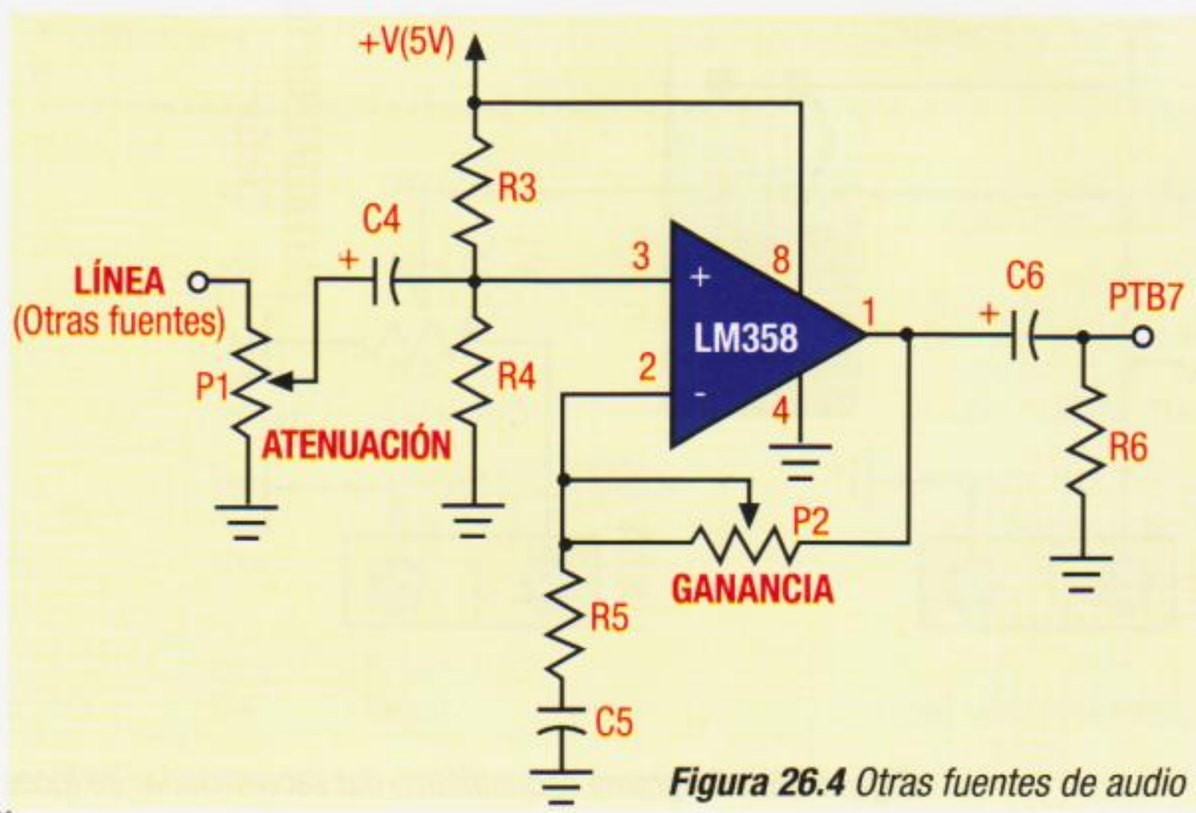


Figura 26.4 *Otras fuentes de audio*

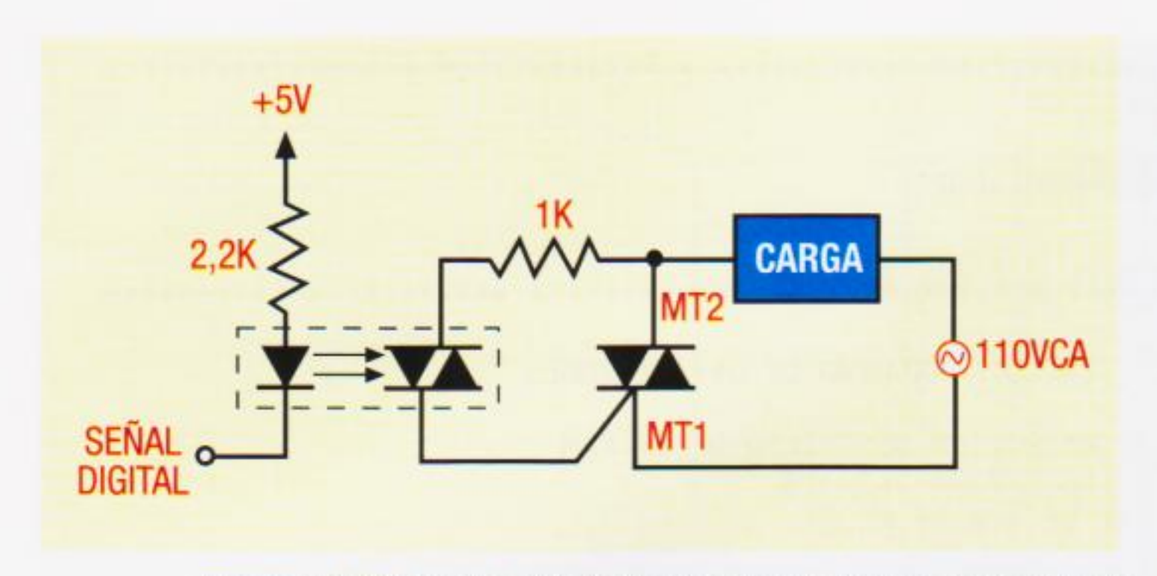


Figura 26.5 *Circuito sugerido para conectar cargas de potencia*

Es posible configurar el convertidor para que haga su trabajo de manera continua, lo que hace que efectúe ciclos consecutivos de conversión A/D sobre el canal de entrada seleccionado, y sobreescribe y actualiza el contenido del registro ADR al concluir cada tiempo de conversión. Para configurar el ADC se deben configurar tres registros para poner a funcionar correctamente el convertidor A/D: el registro de estado y control (ADSCR), el registro de datos (ADR), y el registro de configuración del reloj (ADICLK).

La utilidad práctica de este proyecto se hace más evidente cuando le adicionamos la capacidad de que maneje bombillos o lámparas de buena potencia. En el circuito impreso se ha dejado disponible un conector en donde se puede incorporar para cada salida un circuito como el de la **figura 26.5**, donde se ha utilizado un optoacoplador y un triac para conmutar una carga a un voltaje de110 o 220 VCA.

Descripción del programa

En la **figura 26.6** vemos un diagrama de bloques básico donde podemos apreciar el funcionamiento del programa. En él podemos distinguir los pasos fundamentales como son configurar el registro CONFIG1 y el registro ADCLK de acuerdo a los requerimientos que tengamos, como lo explicaremos en los párrafos que están a continuación. Posteriormente debemos encerrarnos en un ciclo en el cual se envía paso a paso cada uno de los *bytes* que conforman la secuencia y que se encuentran ordenados dentro de una tabla localizada en la memoria de programa.

Entre el envío de cada uno de los *bytes* se debe aguardar un tiempo de espera, el cual a su vez, depende del valor leído en el conversor A/D, el cual proviene de las fuentes de señal o de sonido. Entre mayor sea el valor leído, menor será el tiempo de espera o retardo.

En el programa principal se inicializa el módulo ADC del HC08 cada vez que se reinicia el programa e invoca una subrutina que toma las mediciones del convertidor A/D. Cuando ocurre un estado de *reset*, el primer paso es definir el registro de configuración del *hardware*.

El programa ocupa memoria *flash* y debemos decirle al compilador en qué posiciones de memoria se localizará; para definir estas condiciones utilizamos la directiva "org". En este caso se ha optado que el reloj del ADC siempre funcionará a la misma frecuencia, si no fuera así, deberíamos configurar repetidamente el registro ADICLK.

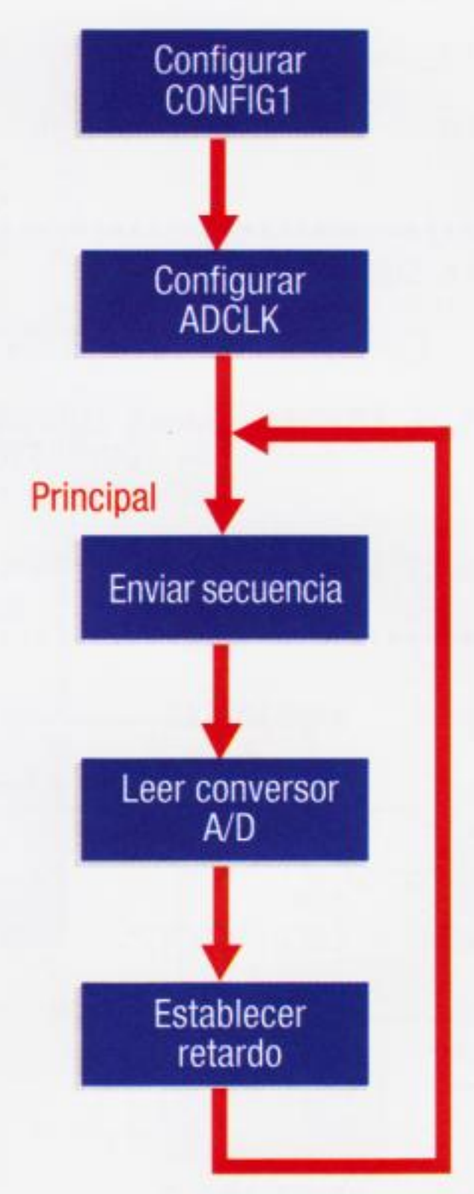


Figura 26.6 *Diagrama de bloques del programa*

```
*****************************************************************************************
*
*                    PROYECTO: LUCES RÍTMICAS
*                    CURSO PRÁCTICO SOBRE MICROCONTROLADORES
*                    FASCÍCULO 32
*                    CEKIT - 2001
*****************************************************************************************

$include 'jl3regsg.inc'                          ;INCLUIR ARCHIVO DE DEFINICIONES

RAM_START            equ    $0080                ;DIRECCIÓN DE INICIO DE LA RAM
FLASH_START_JK3      equ    $EC00                ;SOLO PARA JK3/JL3
RESET_VEC            equ    $FFFE                ; VECTOR DE RESET

        org   RAM_START

dato1                rmb    1                    ;SEPARA UN BYTE
dato2                rmb    1                    ;SEPARA UN BYTE
MEM1                 RMB    1

        org   FLASH_START_JK3                    ;INICIO DE LA MEMORIA FLASH

Reset:
          Rsp
          mov            #%11111111,CONFIG1      ; DESHABILITAR EL COP (WATCHDOG)
          mo v           #%01100000,ADCLK        ; EN ESTE CASO UTILIZAMOS UN CRISTAL DE 8 MHZ
                                                 ; EL CUAL SE DIVIDE POR 8 PARA OBTENER 1MH
          BCLR            7,DDRB                 ; PB7 IN PARA EL ADC
          Bset            6,DDRB                 ; LOS DEMÁS PINES EXCEPTO PB0 SE CONFIGURAN
          Bset            5,DDRB                 ; PARA SER LAS SALIDAS DE LOS DATOS DE LA
          Bset            4,DDRB                 ; SECUENCIA.
          Bset            3,DDRB
          Bset            2,DDRB
          Bset            1,DDRB

Principal:

          clra
          jsr             leer_ad                ; INVOCAR SUBRUTINA DE LECTURA DEL CONVERSOR A/D
          coma                                   ; EN ESTE PUNTO EL DATO LEÍDO SE ENCUENTRA
                                                 ; DISPONIBLE EN EL ACUMULADOR.
          sta             dato1                  ; ÉSTE ES EL RESULTADO, A PARTIR DE ESTE VALOR SE
                                                 ; CONFIGURA EL VALOR DEL RETARDO.
          bra             Principal

*****************************************************************************************
* NOMBRE DE LA SUBRUTINA:  LEER_AD                    ÚLTIMA REVISIÓN: AGOSTO 2001
*
* OBJETIVO:    EJECUTAR UNA CONVERSIÓN A/D DE UNO DE LOS CANALES
*
* CONDICIONES DE ENTRADA: HABER CONFIGURADO ADECUADAMENTE EL REGISTRO ADCLK PARA UNA VELOCIDAD
*                         DE CONVERSIÓN DE 1 MHZ. EL CANAL ADC DESEADO EN EL ACUMULADOR Y EL
*                         RESPECTIVO PIN DISPUESTO COMO ENTRADA.
*
* CONDICIONES DE SALIDA:  EL RESULTADO DE LA CONVERSIÓN EN EL ACUMULADOR
*
*****************************************************************************************
leer_ad
          ora      #%00100111                    ; SELECCIONO MODO DE CONVERSIÓN
                                                 ; CONTINÚO POR EL CANAL 7
          sta      ADSCR
          brclr    COCO,ADSCR,*                  ;ACTIVAR EL ADC Y ESPERAR UNA VUELTA AL CICLO
                                                 ;DE CONVERSIÓN PARA ESTABILIZARSE
          lda      ADR                           ;LIMPIAR EL BIT COCO, LEYENDO EL ADR
          brclr    COCO,ADSCR,*                  ;ESTE TIEMPO TOMA LA LECTURA REAL

          mov      #%00011111,ADSCR              ; TOMAR DATO Y APAGAR EL ADC
          lda      ADR                           ; RETORNA LA MEDICIÓN TOMADA EN EL ACUMULADOR
          rts


          Org      RESET_VEC
          Dw       Reset                         ;al darse reset salta a Start
```

Figura 26.7 Código del proyecto

El lazo principal del programa llama a una subrutina que toma las mediciones con las cuales se establece la velocidad con que se enviará la secuencia. La subrutina maneja todo lo concerniente al convertidor A/D y requiere que el reloj se encuentre activado. De la misma manera requiere de pines de trabajo y de canales de entrada.

El reloj del ADC y los puertos se configuran justo después de un estado de *reset*, de manera que debemos decirle a la subrutina qué canal debe usar. Esto se puede hacer cargando nuestra selección en el acumulador. Cuando la subrutina termina su ejecución el valor medido (con resolución de 8 *bits*) es retornado en el acumulador.

La parte del código de lectura del ADC es un buen ejemplo de reutilización de código, pues esta subrutina puede ser invocada fácilmente desde un programa principal. Por ejemplo, si necesitáramos leer ocho diferentes sensores de temperatura en cada canal, esta rutina podría utilizarse para todas ellas. La única entrada que necesitaría es la identificación del canal AD.

En la porción de código fuente de la **figura 26.7** podemos apreciar la configuración y manejo del conversor análogo digital interno del microcontrolador, de acuerdo a lo expuesto en la sección de teoría. Esta es la porción novedosa dentro de nuestro estudio, la otra parte del código se trata del barrido de los datos de una tabla y la generación de retardos, casos que ya se han explicado en proyectos anteriores con microcontroladores Motorola.

El convertidor A/D puede ser configurado para que funcione en modo simple o en modo continuo, es decir, que haga una sola conversión para que sea chequeado o que permanezca en funcionamiento y se tomen los datos en cualquier momento. Esto se establece mediante el *bit* 5 del registro ADSCR. Ahora, se debe combinar el *bit* de selección de la conversión con la selección del canal de entrada en un valor final de 8 *bits*. La mejor manera de hacer esto es mediante una operación lógica OR.

Por ejemplo, para seleccionar el modo de conversión continua utilizando como entrada el canal 3. Utilizamos el acumulador para pasar a la subrutina el parámetro 3 para el canal de entrada. Si ejecutamos una OR entre el acumulador y el $20 para conversión continua, el resultado es $23.

Tan pronto almacenamos este valor en el ADSCR, el módulo ADC inicia su proceso de conversión. El circuito ADC no se ha estabilizado aún, debemos esperar un ciclo de conversión para tenerlo listo. La manera más rápida de hacer esto es entrar en un ciclo continuo mientras se chequea el *bit* de fin de conversión del registro ADSCR.

Cuando este *bit* está en estado alto, ha finalizado la conversión, lo que quiere decir que el circuito ha tenido tiempo para estabilizarse. Este paso no es necesario si iniciamos la conversión después de que el módulo ADC se ha alimentado.

La lectura del ADR limpia el *bit* COCO. No utilizaremos la primera medición mientras el circuito AD esté inestable, esperaremos la siguiente medición para completar. Apenas se encuentra listo, el *bit* COCO se activa nuevamente y podremos salir del ciclo.

El módulo ADC se desactiva estableciendo los *bits* de selección de canal en el registro ADSCR. Para originar el efecto de luces rítmicas, la medición obtenida en el registro ADR se retorna mediante el acumulador y con él se establece un retardo. Se invoca una tabla para rescatar secuencialmente los datos que se van a enviar por los puertos, aguardando entre cada uno de ellos, el retardo generado por los niveles de las señales de entrada.

Ensamblaje del proyecto

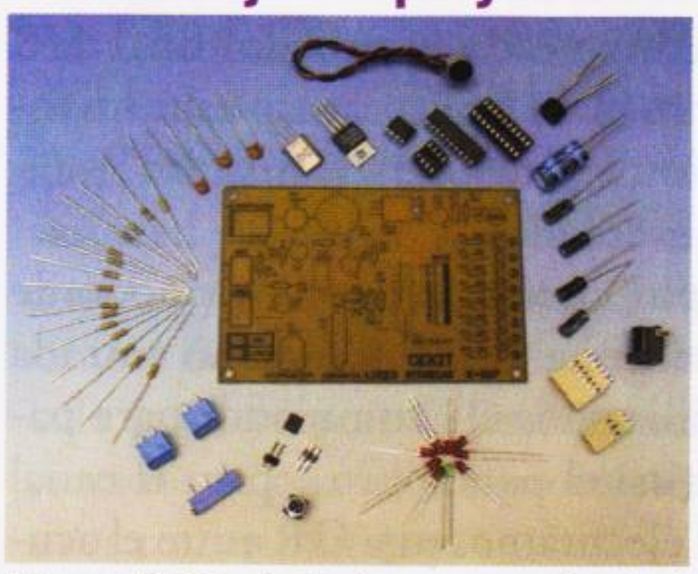
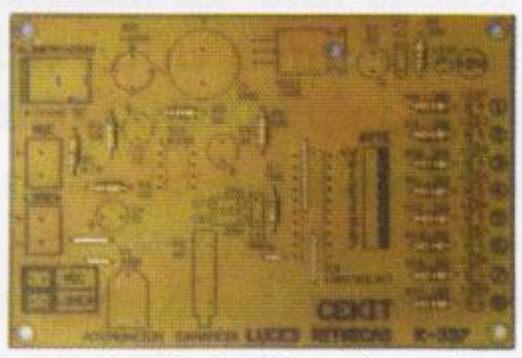


***Figura 26.8** Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el circuito de luces rítmicas CEKIT K-337.*

Con este sencillo proyecto usted podrá divertirse al hacer variar la velocidad de la secuencia de exitación de 8 LEDs en función del sonido; y adicionamente por medio del byte de potencia de CEKIT, se podrán manejar de una manera segura ocho lámparas de hasta 250W cada una. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

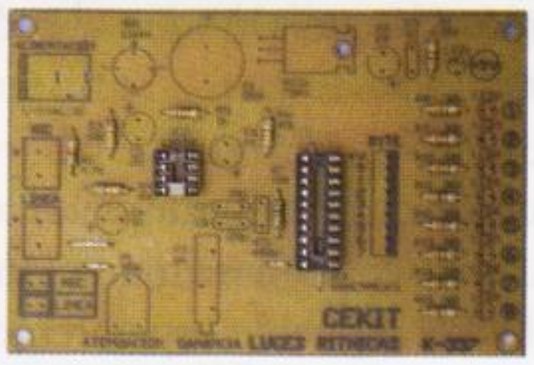
Cant.	Ref.	Descripción
1	IC1	Microcontrolador Motorola 68HC908JK3, programado.
1	IC2	Circuto integrado LM358.
1	REG1	Regulador fijo de +5V -1A, 7805.
1	BR1	Puente rectificador circular de 1A, W04M.
1	LED1	Diodo LED verde de 3mm.
8	LED2 a LED9	Diodos LED rojos de 3mm.
1	C1	Condensador electrolítico de 1000uF - 25V.
2	C2 y C5	Condensadores electrolíticos de 22uF - 25V.
1	C3	Condensador cerámico de 0.1uF - 50V.
2	C4 y C6	Condensadores electrolíticos de 10uF - 25V.
2	C7 y C8	Condensadores cerámicos de 22pF - 50V.
1	XT1	Cristal de quarzo de 4MHz.
1	P1	Reóstato de 100k.
1	P2	Trimmer de 1M.
9	R1, R8 a R15	Resistencias de 300Ω - 1/4W.
1	R2	Resistencia de 4.7k - 1/4W.
3	R3, R4 y R6	Resistencias de 10k - 1/4W.
1	R5	Resistencia de 1k - 1/4W.
1	R7	Resistencia de 1M - 1/4W.
1	K-337	Circuito impreso CEKIT.
2		Conectores de tornillo de 2 pines.
1		Conector en línea de 9 pines M/H.
1		Conector de alimentación AC/DC.
2		Conectores tipo cerca no polarizado de 2 pines.
1		Jumper.
1		Base para circuito integrado de 20 pines.
1		Base para circuito integrado de 8 pines.



Paso 1

Instale y suelde los puentes de alambre y las resistencias de 1/4 de vatio.

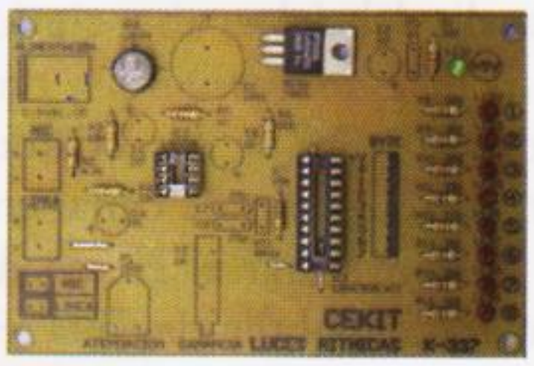
Figura 26.9



Paso 2

Luego instale y suelde las bases para los circuitos integrados IC1 e IC2.

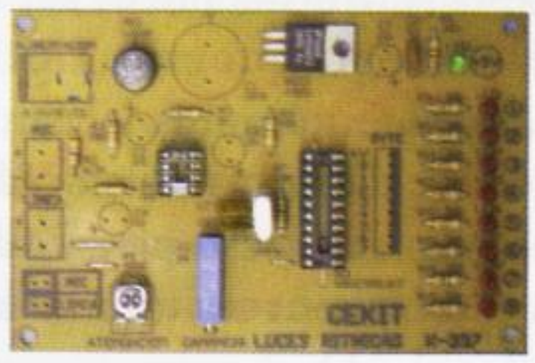
Figura 26.10



Paso 3

Instale y suelde el puente rectificador BR1, el regulador de voltaje integrado REG1, y los nueve diodos LED de 3mm.

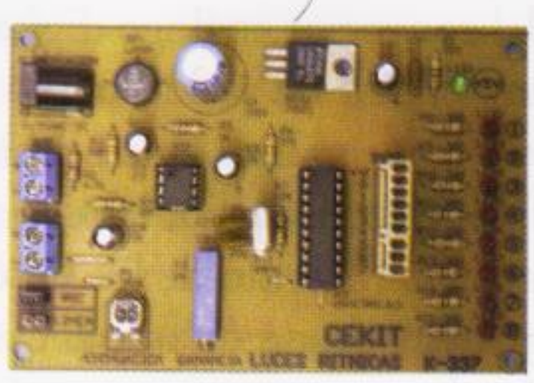
Figura 26.11



Paso 4

Instale y suelde el TRIMMER P2, el reóstato P1, los condensadores cerámicos, y el cristal de cuarzo XT1.

Figura 26.12



Paso 5

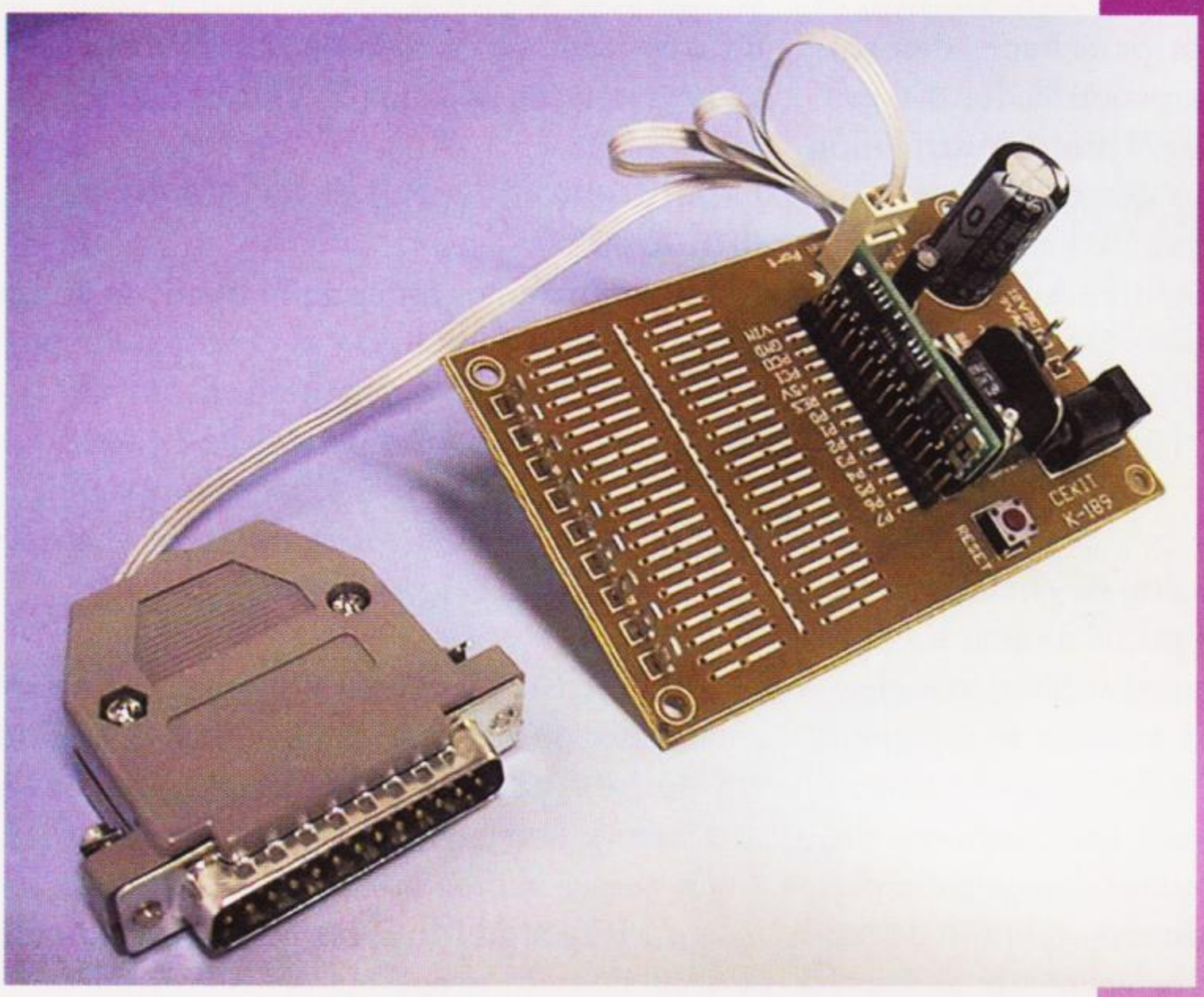
Aspecto final del circuito de las luces rítmicas K-337, después de instalar y soldar los condensadores electrolíticos, los conectores para la entrada de sonido (micrófono y línea), el conector para la alimentación, los conectores tipo cerca para el selector, el conector para el byte de potencia, e insertar los respectivos circuitos integrados.

Figura 26.13

Paso 6. Pruebas finales

Alimente el circuito mediante un adaptador de voltaje. Antes de insertar el microcontrolador pruebe que el voltaje de entrada debido a las señales del micrófono o de la otra fuente de señal, se encuentren dentro del rango adecuado y el circuito no corra peligro; para este propósito la tarjeta cuenta con dos resistencias variables que permiten ajustar los niveles de la señal.

PROYECTO 27



HERRAMIENTAS DE DESARROLLO PARA BASIC Stamp I

Los microcontroladores BASIC Stamp son los dispositivos, dentro de su clase, más fáciles de manejar que existen en el mercado. En esta ocasión, vamos a construir un módulo que nos permitirá poner en practica un gran número de experimentos y nos facilitará enormemente el desarrollo de sistemas y soluciones reales.

En la sección de teoría hemos analizado los conceptos básicos de los microcontroladores BASIC Stamp, como su arquitectura y conjunto de instrucciones. En la sección de programación llevamos a cabo algunos ejercicios partiendo de los conceptos teóricos. Ahora, estudiaremos una serie de herramientas que nos permitirán adentrarnos en el mundo del desarrollo electrónico con estos fabulosos microcontroladores.

Tarjeta para desarrollo de prototipos para BASIC Stamp I. CEKIT K-189.

La primera tarjeta que estudiaremos en esta ocasión es la tarjeta para el desarrollo de prototipos con BASIC Stamp I. Este circuito es básicamente una tarjeta para hacer desarrollos finales con este microcontrolador. En este módulo se puede colocar el microcontrolador para su programación ya que tiene lista la conexión para el cable que va hasta el puerto paralelo de la computadora. Además, como posee una fuente de alimentación regulada, no existe ningún riesgo para el microcontrolador ni para los otros elementos que se utilicen en el diseño.

El sistema tiene una área de circuito impreso libre donde el usuario puede montar los otros elementos que conforman el sistema, por ejemplo, transistores, diodos, resistencias, circuitos integrados, etc. Los pines del microcontrolador se pueden llevar fácilmente hasta los otros elementos ya que poseen puntos de conexión para ello. También, se ha dispuesto un botón de *reset* para que el usuario pueda fácilmente acceder a dicha función. En la **figura 27.1** se muestra el diagrama esquemático del circuito y en la **figura 27.2**, el circuito impreso.

La principal ventaja de este circuito es que en él se pueden incorporar gran cantidad de elementos con los que el microcontrolador puede interactuar para llevar a cabo tareas específicas. La fuente de alimentación del sistema puede ser un adaptador de corriente o un transformador que entregue en su salida 9VCA ó 12 VCC.

Experimentando con la tarjeta de prototipos

Ahora mostraremos un ejemplo donde se ilustra la metodología de trabajo y el desarrollo de proyectos con la tarjeta de prototipos K-189 de CEKIT. Vamos a desarrollar un sencillo ejemplo que nos servirá para comprobar el buen funcionamiento de la tarjeta y para familiarizarnos con el uso de esta sencilla pero práctica herramienta.

El objetivo del ejercicio es desarollar un secuenciador de luces básico en el cual, el sentido con que se se encienden los cuatro diodos LED, depende de la señal dada por un pulsador (S1). Cada que el microcontrolador detecta una señal en S1, se modifica el sentido de rotación de la secuencia. El programa de este ejercicio lo podemos apreciar en la **figura 27.3** y con el diagrama esquemático de la **figura 27.4** podemos guiarnos en este fácil ejercicio. Los diodos

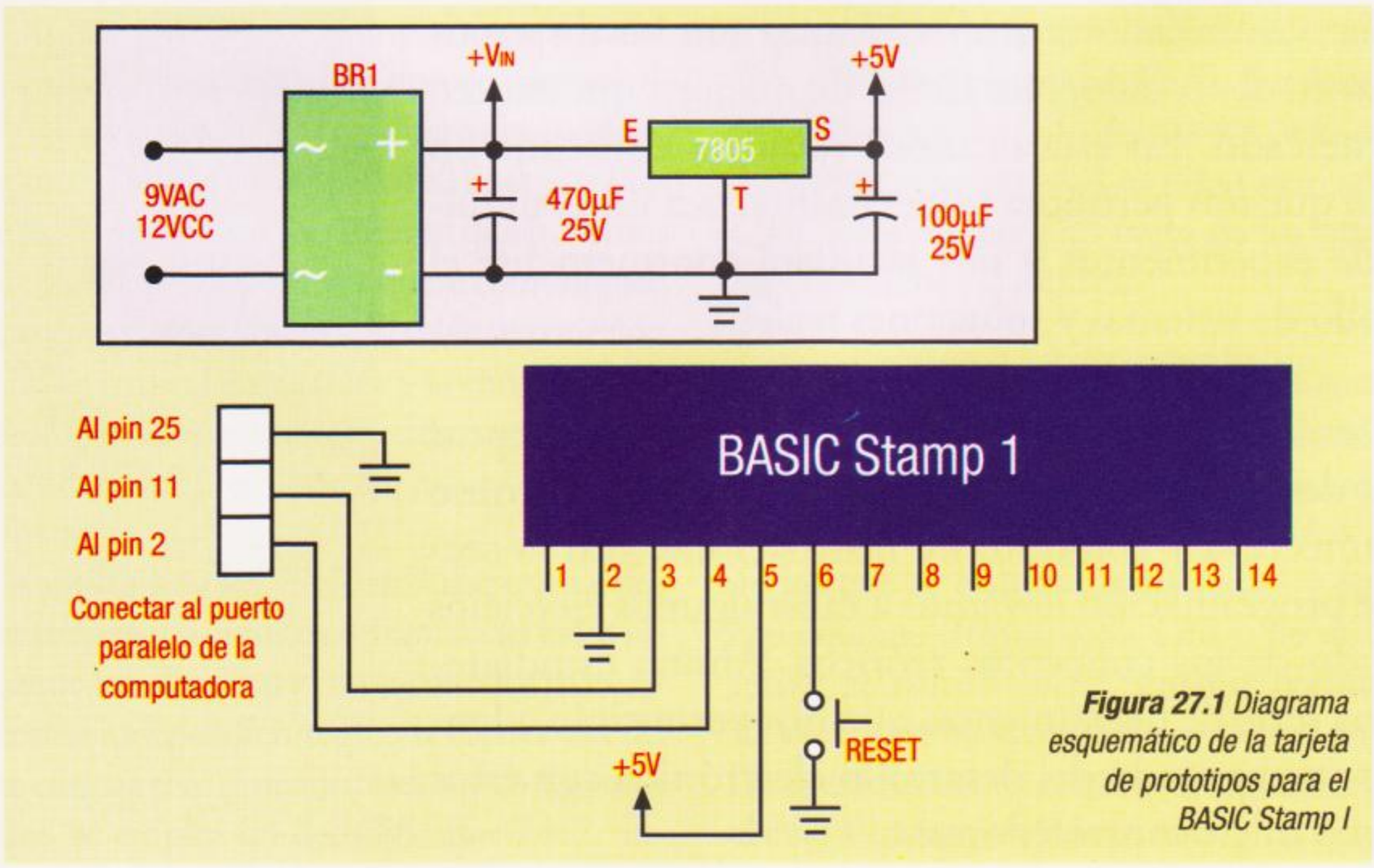


Figura 27.1 *Diagrama esquemático de la tarjeta de prototipos para el BASIC Stamp I*

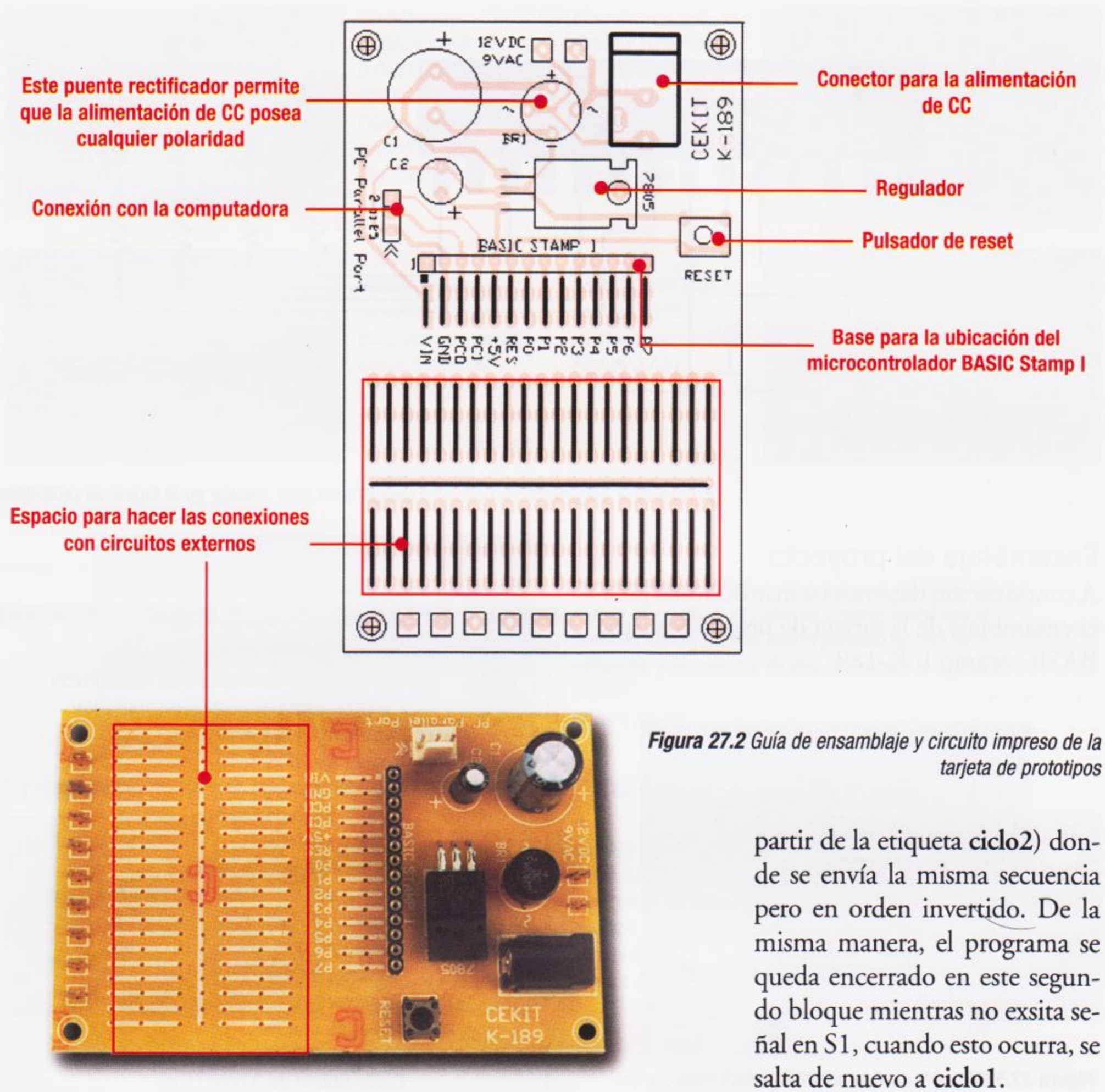


Figura 27.2 *Guía de ensamblaje y circuito impreso de la tarjeta de prototipos*

LED se han conectado a los pines P1, P3, P5 y P7 del BASIC Stamp I por medio de cuatro resistencias de 470 Ohmios y el pulsador S1 se ha conectado al pin P0. De las **figuras 27.13** a la **27.16** se muestra paso a paso cómo instalar los circuitos sobre la tarjeta de prototipos.

Todas las instrucciones que lo componen ya se han analizado en las secciones de teoría y programación. En la primera línea se configura el funcionamiento de los puertos y luego se pueden distinguir básicamente dos bloques, cada uno origina una secuencia distinta sobre el puerto del BASIC Stamp. En el primer bloque (a partir de la etiqueta **ciclo1**) podemos apreciar que por el puerto se envía la secuencia 8,4,5,2,1 y el proceso se repite mientras no exista una señal en S1. Cuando esto ocurre el programa salta al segundo bloque (a partir de la etiqueta **ciclo2**) donde se envía la misma secuencia pero en orden invertido. De la misma manera, el programa se queda encerrado en este segundo bloque mientras no exsita señal en S1, cuando esto ocurra, se salta de nuevo a **ciclo1**.

```
let dirs = %11111110
ciclo1:
        for b2 = 0 to 3
        LOOKUP
b2,(189,167,222,127),b3
        let pins = b3
        pause 700
        if pin0 = 0 then ciclo2
        next
        GOTO ciclo1
ciclo2:
        for b4 = 0 to 3
        LOOKUP
b4,(127,222,167,189),b5
        let pins = b5
        pause 700
        if pin0 = 0 then ciclo1
        next
        GOTO ciclo2
```

Figura 27.3 *Código del ejercicio para montar en la tarjeta de prototipos*

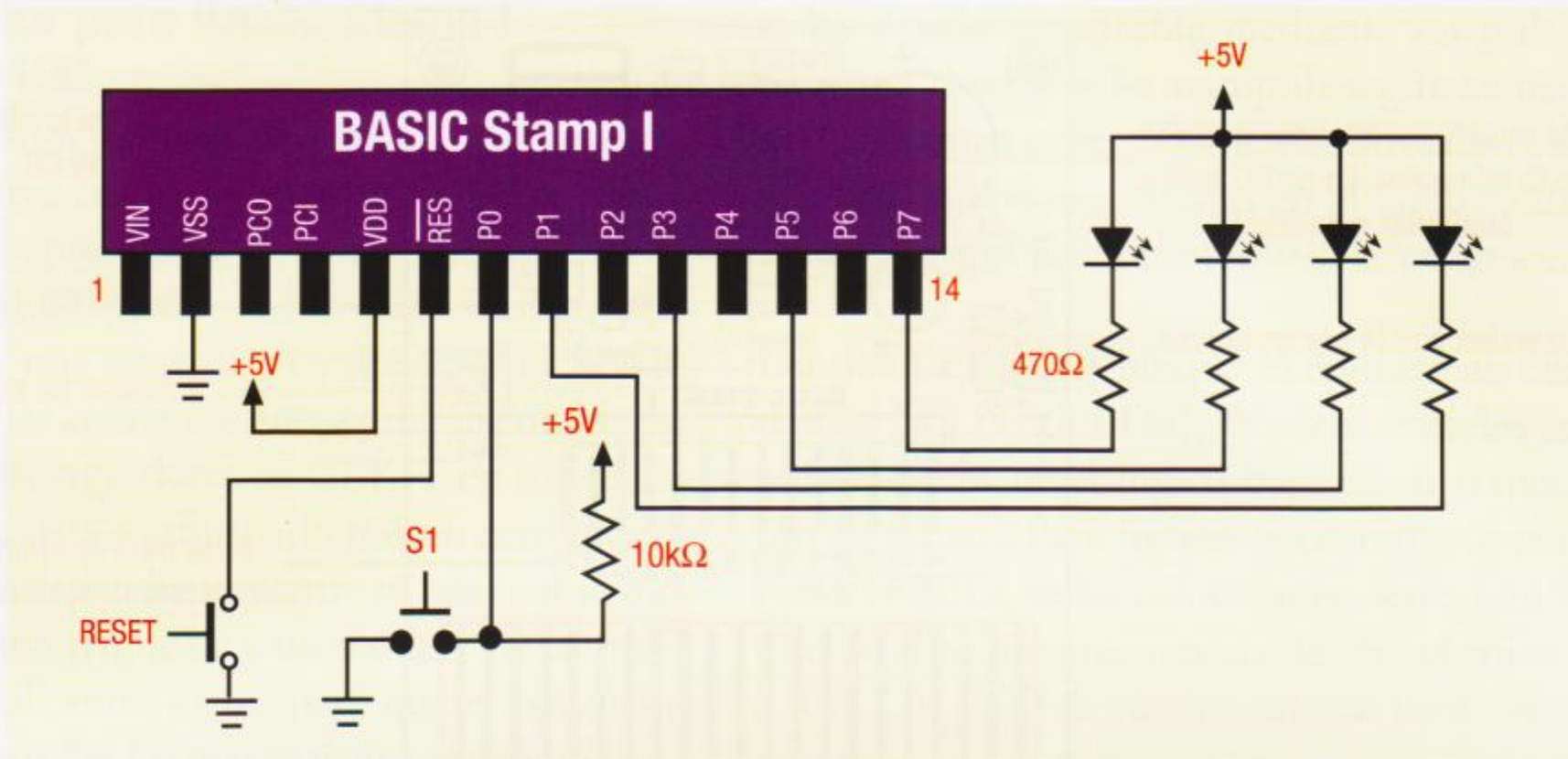


Figura 27.4 *Diagrama esquemático del ejercicio para montar en la tarjeta de prototipos*

Ensamblaje del proyecto

A continuación daremos las instrucciones para el ensamblaje de la tarjeta de prototipos para BASIC Stamp I, K-189

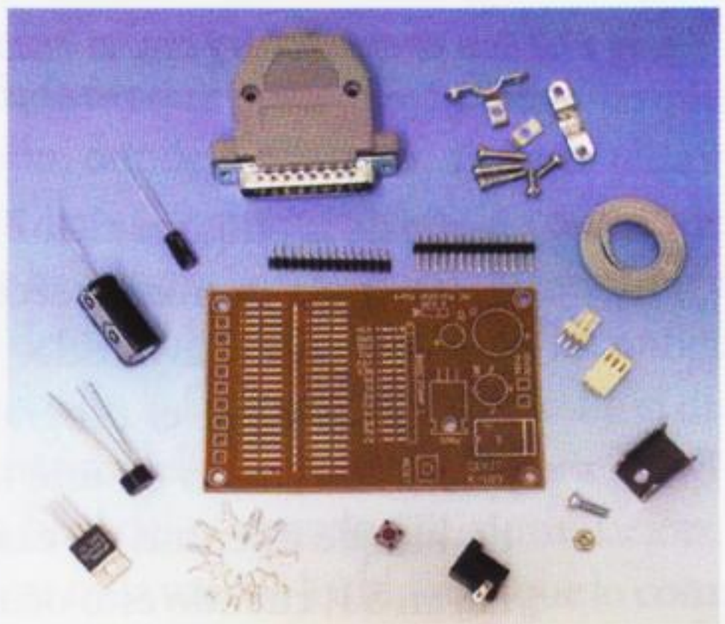
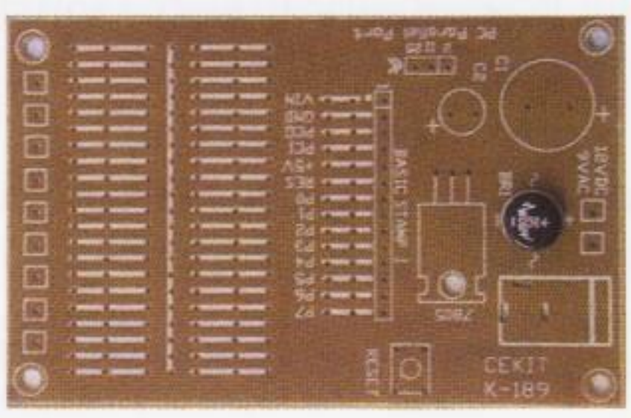


Figura 27.5 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar la tarjeta de prototipos del BASIC Stamp I, K-189*

LISTA DE MATERIALES

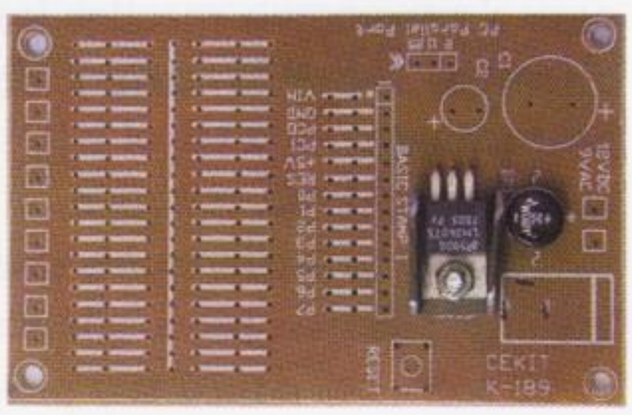
Cant.	Ref.	Descripción
1	C1	Condensador electrolítico de 1000µF/25V
1	C2	Condensador electrolítico de 10µF/25V
1	BR1	Puente rectificador de 1 A
1	7805	Regulador 7805
1	RESET	Pulsador pequeño de 4 patas para impreso
1	K-189	Circuito impreso Cekit
1		Conector CA/CC
1		Socket en línea de 14 pines
11		Terminales (espadines) para circuito impreso
1		Tornillo milimétrico 3x7 con tuerca
1		Conector DB25 macho con carcaza
1		Conector en línea de 3 pines
1		Conector tipo cerca no polarizado de 14 pines (de una fila)
1		Disipador TO-220
1		Cable ribbon de 3 hilos (1 m)

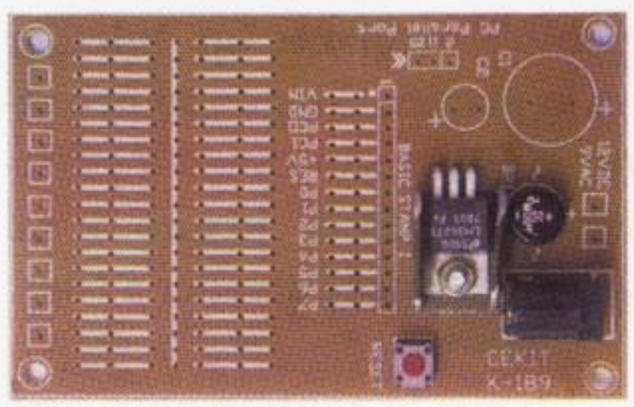
Esta tarjeta se constituye en una herramienta muy útil para el usuario, a la hora de desarrollar, programar y monitorear proyectos experimentales con el microcontrolador BASIC Stamp I. Cuenta con un regulador de voltaje de +5V y 1A que puede ser utilizado para alimentar los circuitos que en ésta se construyan. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:



Paso 1
Instale y suelde el puente rectificador circular BR1. Éste se utiliza para proteger el microcontrolador de la alimentación con polaridad invertida.
Figura 27.6

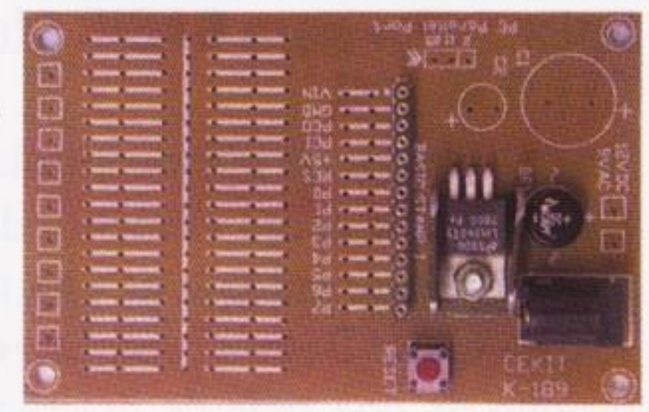
Paso 2
Instale y suelde el regulador fijo de voltaje de +5V que alimenta a todo el circuito. Móntelo sobre un disipador tipo TO-220.
Figura 27.7

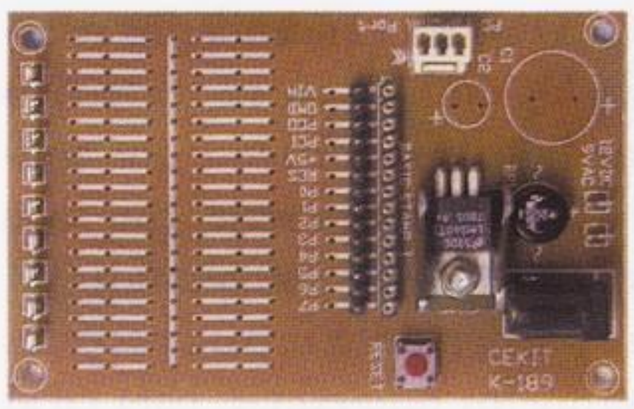




Paso 3
Instale y suelde el pulsador de reset y el conector de entrada CA/CC para la alimentación.
Figura 27.8

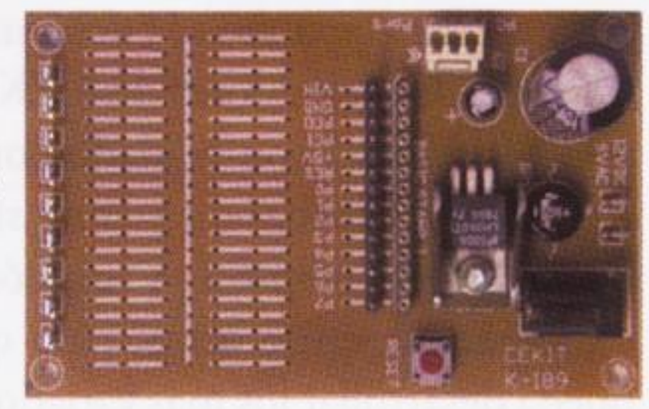
Paso 4
Instale y suelde el socket en línea de 14 pines que soporta al microcontrolador BASIC Stamp I.
Figura 27.9

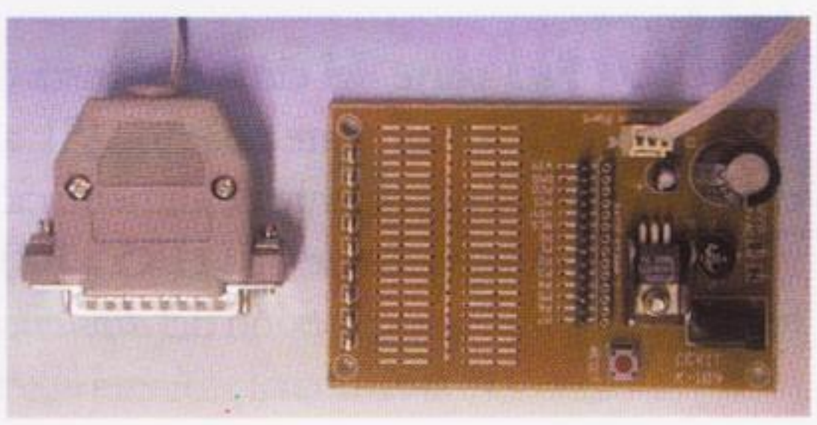




Paso 5
Instale y suelde el conector tipo cerca no polarizado en paralelo con el socket del Stamp I, los espadines y el conector de tres pines para el cable de programación.
Figura 27.10

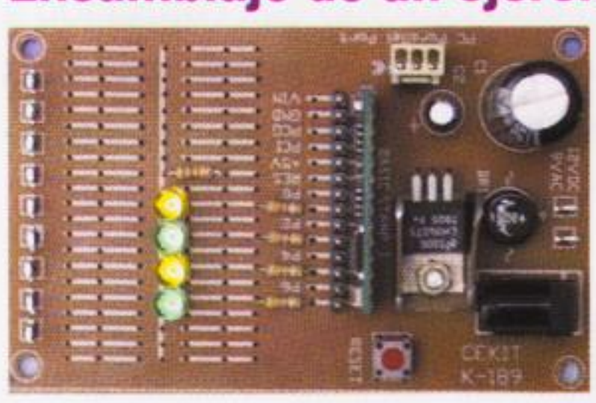
Paso 6
Instale y suelde los condensadores electrolíticos C1 y C2.
Figura 27.11





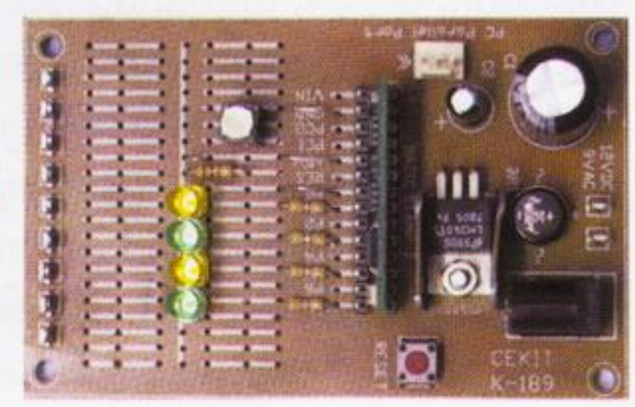
Paso 7
Aspecto final de la tarjeta de prototipos del BASIC Stamp I luego de construir el cable de programación con el conector DB-25 para el puerto paralelo en uno de sus extremos, y el conector de tres pines para la tarjeta en el otro.
Figura 27.12

Ensamblaje de un ejercicio en la tarjeta de prototipos



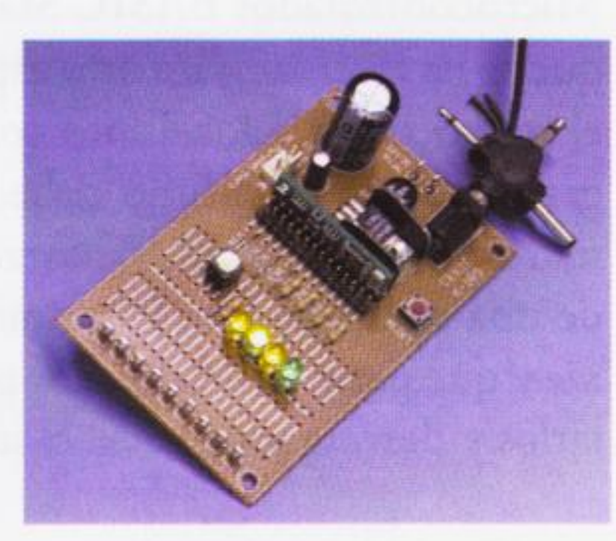
Paso 1
Instale y suelde primero las resistencias y los diodos LED y haga las conexiones con puentes de alambre que se requieran de acuerdo con el diagrama esquemático del ejercicio (figura 27.4).
Figura 27.13

Paso 2
Instale y suelde el pulsador y su resistencia en serie de acuerdo al diagrama esquemático.
Figura 27.14



Paso 3
Una vez terminado de ensamblar el circuito de este ejercicio, inserte el microcontrolador Stamp I en su socket, y conecte el cable de programación a la tarjeta de desarrollo. Luego de haber desarrollado y depurado el programa, transfiéralo hacia la memoria de programa EEPROM del sistema.
Figura 27.15

Paso 4
Una vez transferido el programa de aplicación se puede desconectar el cable de programación. Para ejecutar el programa, basta con conectar la alimentación al sistema.
Figura 27.16



Paso 8. Pruebas finales
Si todos los elementos se encuentran en buen estado y la alimentación se encuentra dentro del rango indicado, la tarjeta debe funcionar sin problemas. La manera más sencilla de probar su funcionamiento es montando cualquiera de los ejercicios mostrados en esta sección y en la de programación. Una vez escrito el programa en el entorno de programación pulse CTRL-R y observe y disfrute el funcionamiento de este maravilloso microcontrolador.

Entrenador para BASIC Stamp I CEKIT K-190

Un entrenador de microcontroladores es un aparato que sirve como elemento de soporte para el proceso de aprendizaje de esta tecnología. El K-190 de CEKIT es un producto que facilita el desarrollo de una serie de prácticas con el BASIC Stamp I. Este aparato se encuentra disponible patra todos los seguidores de CEKIT en toda Latinoamérica. Está conformado por un conjunto de circuitos, independientes entre sí, tales como diodos LED, interruptores y un módulo de comunicación serial, entre otros, para que el estudiante pueda desarrollar los más variados experimentos y proyectos sencillos (para los que se inician), o de cierta complejidad (para los más experimentados).

Cada bloque tiene listas las conexiones a la fuente de alimentación y a los diferentes elementos que lo componen; de esta forma, sólo se requiere conectar las entradas y/o salidas al sitio que corresponda.

Cada bloque posee unos pines o terminales de conexión, donde, utilizando alambre telefónico, se pueden llevar algunas señales de control desde un bloque hasta otro, permitiendo que según el circuito que se desee probar, el usuario haga la configuración necesaria. En la **figura 27.17** se muestra el diagrama esquemático del circuito y en la **figura 27.18** la apariencia final del entrenador. A continuación, vamos a explicar cada uno de los bloques que lo componen.

Microcontrolador BASIC Stamp I. En este bloque se ha reservado un área especial para colocar el microcontrolador, junto con el cable de programación. Este último debe ir desde los pines marcados en el circuito, hasta el puerto paralelo de una computadora PC, donde se tiene el *software* que permite escribir los programas, compilarlos y descargarlos hacia el microcontrolador.

También, se han dispuesto unos conectores especiales que sirven para extender los pines del microcontrolador; de esta forma, las señales de entrada y salida, se pueden llevar hasta los otros bloques del circuito. Por su parte, el pin de *reset* se ha dejado disponible mediante un pulsador para que el usuario lo manipule según su necesidad. Los otros pines del microcontrolador ya tienen sus conexiones hechas en el circuito de tal manera que el usuario no puede acceder a ellos.

Diodos LED. Este bloque lo conforman cuatro diodos LED (D1 a D4). Para encenderlos se requiere de un nivel lógico bajo en su respectivo pin de acceso. Para limitar la corriente que consume el LED, se ha colocado en serie, con cada uno de ellos, una resistencia de 560 ohmios (R1 a R4). Los LED se pueden utilizar para hacer las prácticas en los primeros programas, para crear secuencias y como monitores de estados lógicos.

Dipswitch. Esta parte del circuito consta de cuatro interruptores normalmente abiertos (S1 a S4), los cuales se conectan a la fuente de alimentación positiva mediante resistencias de 1KΩ (R5 a R8). De esta forma, en los pines de conexión del *dipswitch* se tiene un nivel lógico alto si el interruptor está abierto, pero, cuando el interruptor se cierra, se obtiene un nivel lógico bajo ya que su otro extremo está conectado a tierra. Se pueden utilizar en las primeras prácticas para experimentar la lectura de los puertos, para simular sensores, o entradas de control.

Pulsadores. Son interruptores pulsadores normalmente abiertos (S5 y S6), conectados a la fuente de alimentación con una resistencia de 1KΩ (R9 y R10). En los pines de conexión se tiene normalmente un nivel lógico alto, pero cuando se oprimen los botones se obtiene un nivel lógico bajo. Son de gran utilidad cuando se practica la lectura de puertos o cuando se requiere simular entradas de sensores u otros dispositivos.

Salida con relé. Para manejar cargas de potencia desde el microcontrolador podemos emplear un relé, el cual se opera a través de un transistor NPN. Para activarlo, sólo se debe poner un nivel lógico alto en el pin marcado IN. Dentro de las posibles aplicaciones de este bloque se encuentran los temporizadores, las alarmas, el manejo de aparatos de corriente alterna, etc.

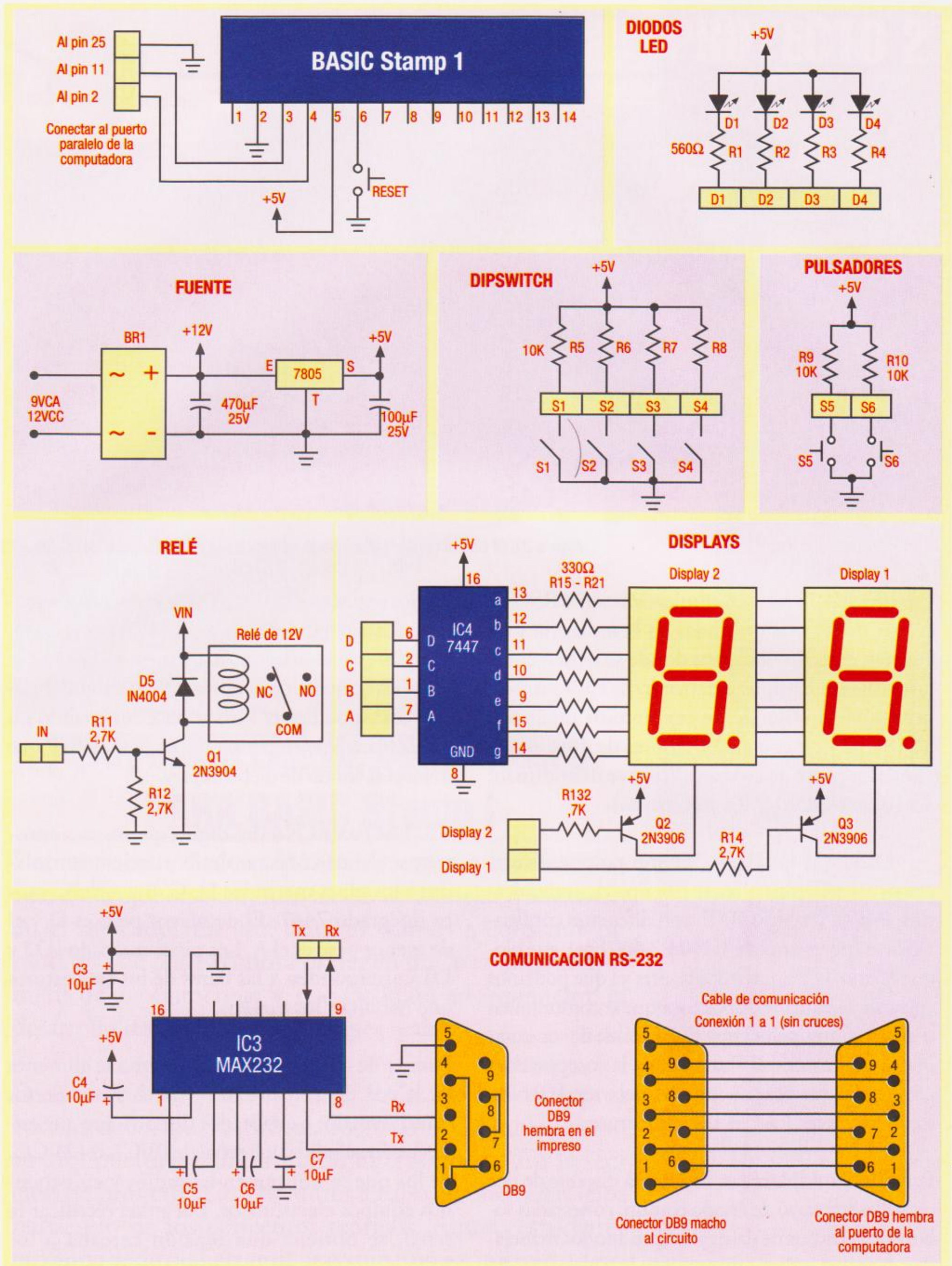


Figura 27.17 *Diagrama esquemático del entrenador para BASIC Stamp I*

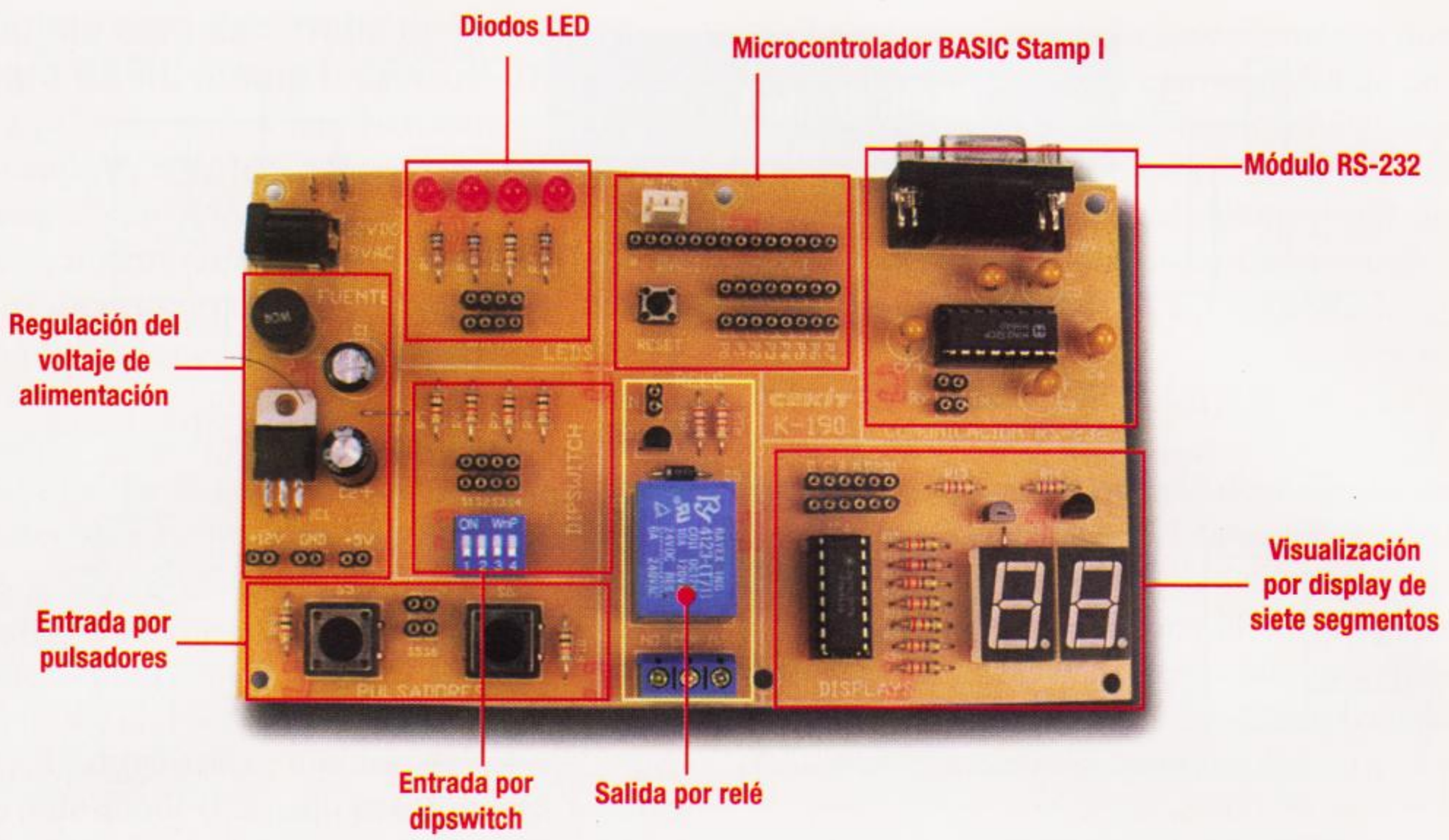


Figura 27.18 *Conjunto de módulos pertenecientes al entrenador para BASIC Stamp 1*

Interfaz RS-232. La comunicación serial RS-232 es una de las más utilizadas en la industria y en general, en todos los sitios donde se tienen computadoras o equipos electrónicos. Para establecerla sólo se requiere de un circuito integrado MAX232 y cinco condensadores de 10µF (tantalio); los pines de comunicación se denominan: Tx (transmisión) y Rx (recepción).

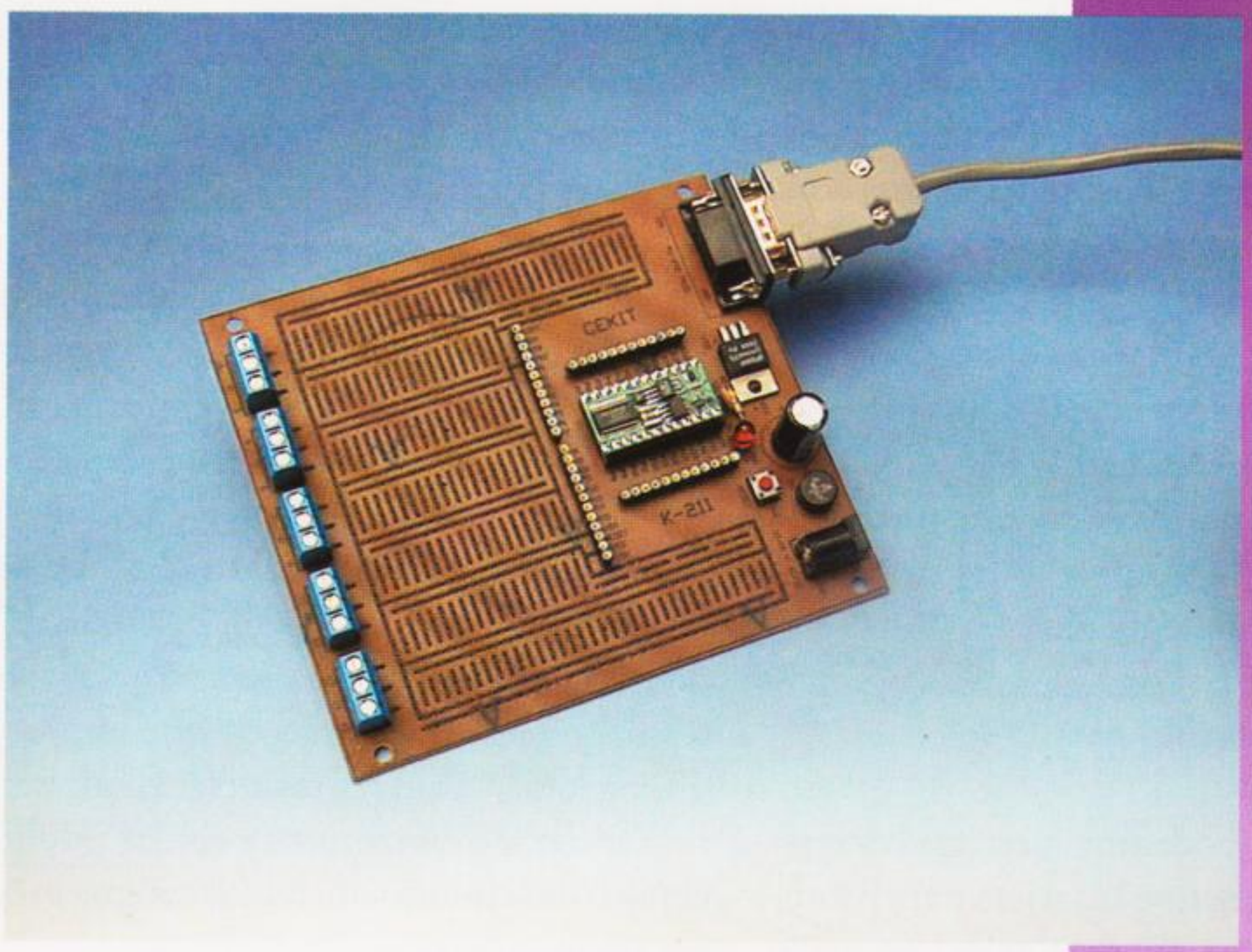
Dado que los BASIC Stamp poseen instrucciones especiales para hacer este tipo de comunicación (SERIN y SEROUT) con diferentes configuraciones (por ejemplo a 1.200 y 2.400 bps), este bloque es uno de los más importantes ya que podemos construir fácilmente dispositivos que se comuniquen con una computadora tipo PC. El cable de conexión que se utilice entre el entrenador y la computadora o equipo al que nos conectamos, debe ser de 9 hilos y con los cables 1 a 1, es decir, sin cruces.

Pantallas multiplexadas. El sistema dispone de dos pantallas (*displays*) de ánodo común, conectadas sobre el mismo bus de datos y con los ánodos manejados de forma independiente. Para la conversión del dato que entrega el microcontrolador al código de 7 segmentos, se emplea un decodificador 7447. Todos los segmentos de los *displays* se encuentran conectados a los pines de salida del 7447; por lo tanto, el dígito que se desea mostrar en ese momento, se debe seleccionar mediante la habilitación del *display* correspondiente. Esto se hace conectando un nivel lógico bajo en la base del transistor PNP que maneja el ánodo de dicho *display*.

Los cuatro *bits* del dígito que se va a mostrar, se deben conectar desde el microcontrolador a los pines marcados D, C, B y A del circuito integrado 7447. El de mayor peso es D y el de menor peso es el A. Los pines marcados D2 y D1 corresponden a las bases de los transistores que habilitan los *displays*.

Fuente de alimentación. El sistema se alimenta de la red de 110 ó 220 VCA, a través de un transformador o adaptador de corriente que entregue en su salida una señal de 9VCA ó 12VCC, de los que se utilizan en las radios y en pequeños equipos electrónicos. Luego de rectificar la señal, se obtiene una tensión cercana a los 11VCC que se emplea para alimentar el relé. De igual forma, se conecta el regulador de +5V, el cual alimenta toda la parte digital del sistema.

PROYECTO 28



TARJETA PARA PROTOTIPOS CON BASIC Stamp II

Con este sistema se pueden llevar a cabo todas las pruebas pertinentes para el desarrollo de un prototipo final que incluya un microcontrolador BASIC Stamp II. Esta tarjeta se acomoda a muchas aplicaciones prácticas y puede usarse como producto final incluyendo los principales elementos del circuito desarrollado.

Gracias a su versatilidad y a la gran cantidad de funciones que poseen los microcontroladores, han revolucionado la forma de diseñar productos electrónicos. Esto ha hecho que los ingenieros, técnicos y aficionados a la electrónica vuelvan sus ojos hacia tan maravillosa tecnología. Por tal razón, el desarrollo de herramientas y sistemas que permitan hacer prácticas con estos dispositivos adquiere gran importancia; además, es una labor fundamental facilitar e incentivar la capacitación en esta área.

Descripción general

La tarjeta para prototipos con BASIC Stamp II consta de los elementos básicos que deben acompañar al microcontrolador y de una sección de circuito impreso de tipo universal, es decir, con perforaciones y pistas que pueden ser utilizadas para el ensamblaje de diferentes tipos de circuitos con elementos comunes, tales como circuitos integrados, resistencias, condensadores, diodos, etc.

Características

- Dimensiones: 120 mm x 130 mm
- Alimentación: 9-18 VCC ó 9-15 VCA
- Programación: serial RS-232 por medio de conector DB9
- Indicador de voltaje regulado
- Pulsador de *RESET*
- Cable de comunicación para PC

En la **figura 28.1** podemos apreciar el diagrama de bloques que describe la tarjeta para prototipos. Obviamente que el corazón del proyecto es el microcontrolador como tal. La fuente de alimentación, tanto para el microcontrolador como para la sección de impreso universal, está basada en un rectificador de onda completa y un regulador de 5 voltios. Observe en la **figura 28.2** el diagrama esquemático de nuestra tarjeta.

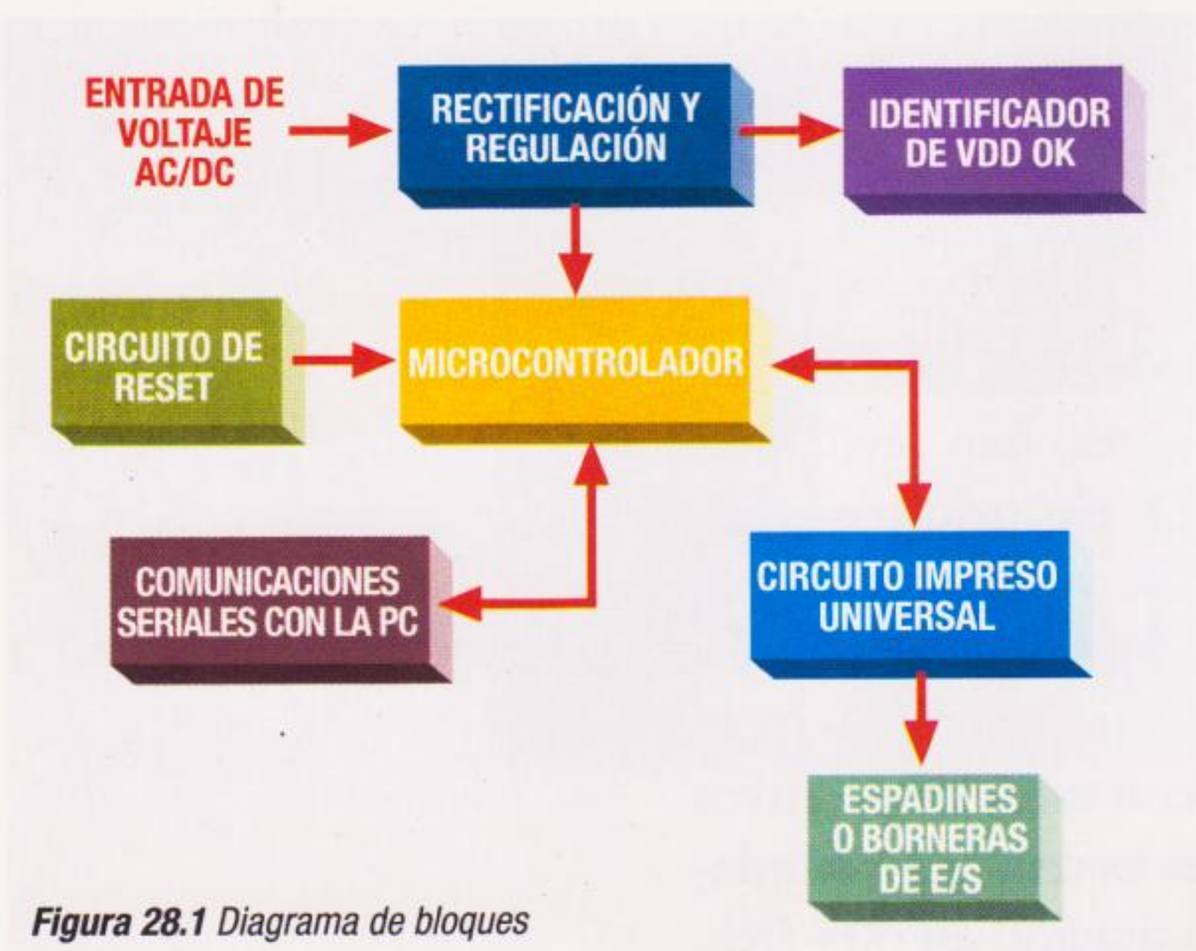


Figura 28.1 *Diagrama de bloques*

Dado que el regulador también alimentará el circuito del prototipo, se debe tener cuidado de no exceder su capacidad de corriente. De por sí, los elementos que ya van en la tarjeta tienen un consumo de 20 mA, por lo tanto, los dispositivos que se ensamblen en el impreso universal no deben consumir más de 20 mA. Si necesita más corriente deberá instalar un disipador a este regulador.

El circuito también contiene un pulsador de *RESET*, etiquetado como SW1, con el fin de inicializar el microcontrolador en forma manual.

Con el objeto de tener disponibilidad de cada uno de los pines del microcontrolador, la tarjeta posee dos juegos de conectores hembra que permiten el uso de puentes a diferentes partes del circuito. En caso de que el prototipo se quiera dejar como producto final, se pueden retirar dichos conectores y soldar puentes hacia el resto del circuito, de tal forma que la conexión sea permanente.

Dentro de la sección de circuito impreso universal hemos construido buses de alimentación a través de todo el espacio del mismo. Cada uno de los buses lleva los 5 voltios provenientes del regulador, marcados como VDD y la tierra general del circuito, marcada como VSS. Con este sistema se facilita enormemente la alimentación a otros circuitos que construyamos dentro de la tarjeta de prototipos.

Ensamblaje

Para el ensamblaje de la tarjeta utilice la guía de montaje de la figura 28.3. Recuerde que las soldaduras deben quedar pequeñas, brillantes y uniendo perfectamente el pin del elemento con el cobre del circuito impreso.

Lo primero que se debe soldar son los componentes de menor altura, ojalá en el siguiente orden: la resistencia R1,

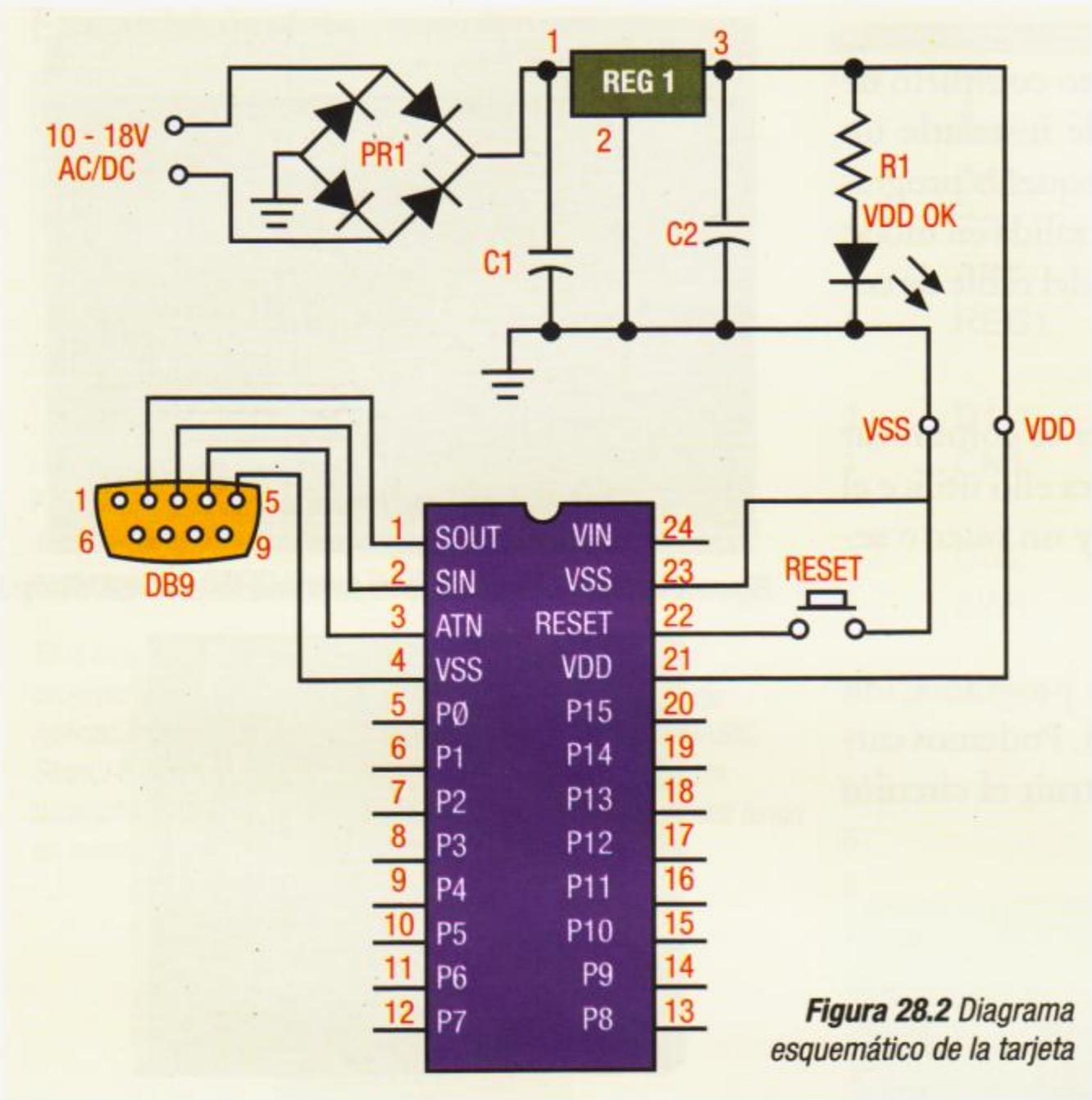


Figura 28.2 *Diagrama esquemático de la tarjeta*

recer el LED1 encendido indicando que el voltaje está correcto. Sin embargo, verifique que tenga un valor entre 4,85 y 5,15 voltios ya que este proviene del regulador.

El cable para comunicar la tarjeta con la PC debe ser 1 a 1, es decir sin cruces. En la **figura 28.4** podemos apreciar la forma de elaborar dicho cable.

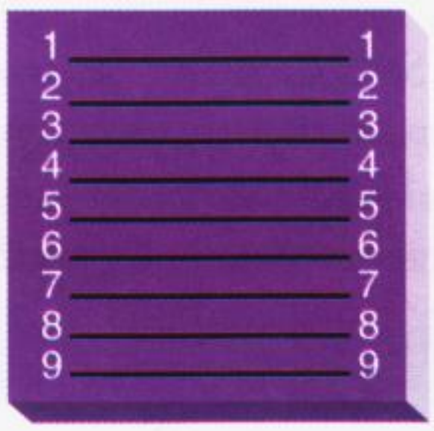


Figura 28.4 *El cable de conexión debe ser uno a uno*

el condensador cerámico C2, el puente rectificador PR1, el pulsador SW1 y la base de 24 pines para el microcontrolador. Luego suelde los componentes de mayor tamaño en el siguiente orden: el diodo LED D1, el regulador REG1, los conectores CN1 y CN2, y por último el condensador C1.

Antes de alimentar el circuito es recomendable hacer pruebas de continuidad con un multímetro para corroborar que no haya algún cortocircuito, especialmente entre las pistas que transportarán voltajes a diferentes puntos del circuito.

Una vez que se hayan hecho las pruebas anteriores, puede energizar el circuito. Deberá apa-

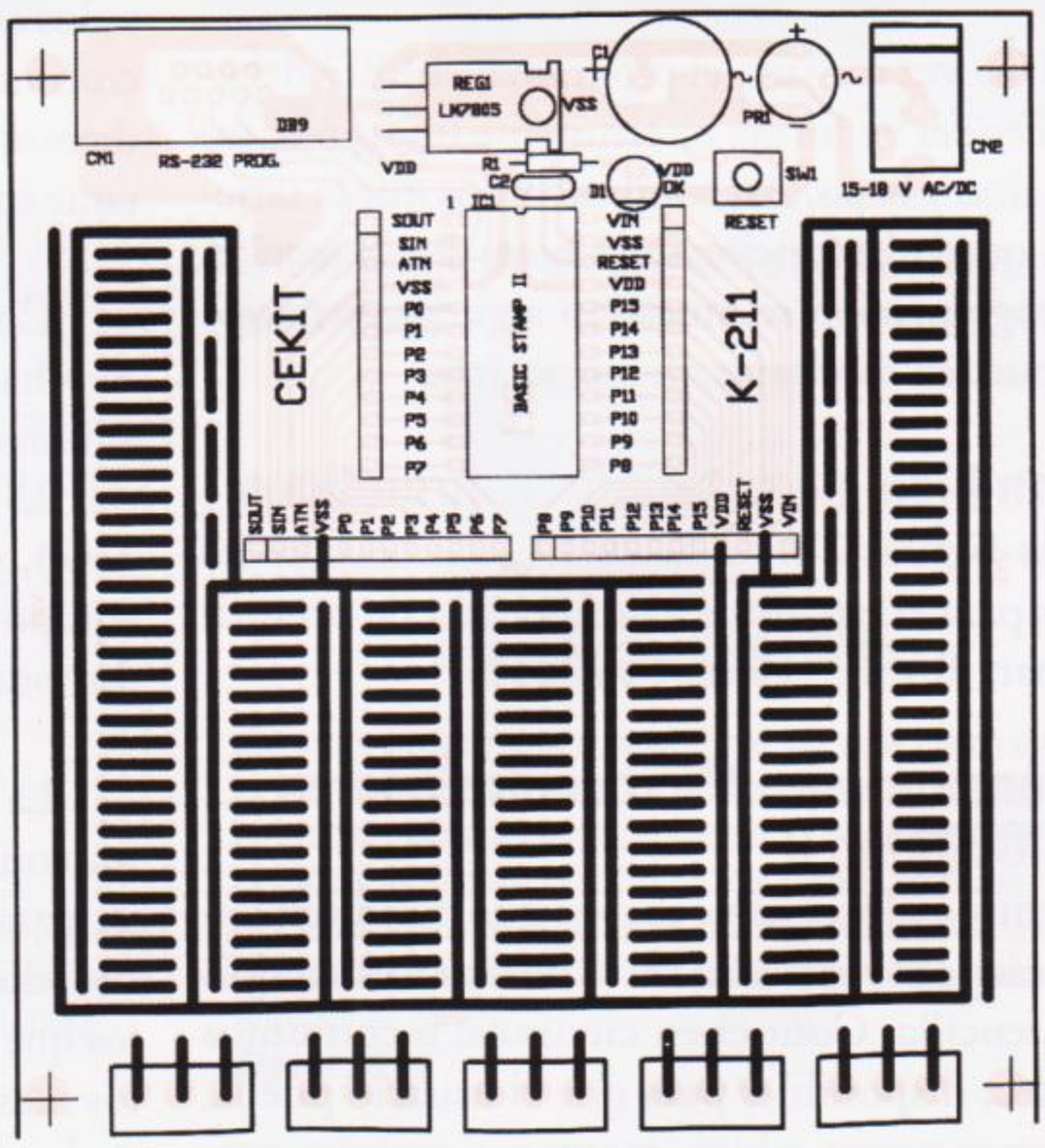


Figura 28.3 *Guía de montaje*

Pruebas de programación

Para comprobar el funcionamiento completo de la tarjeta es necesario, además de instalarle un microcontrolador, elaborar un pequeño programa que utilice al menos un pin de salida en modo intermitente y transmitir a través del cable de comunicación serial.

Lo primero que se debe hacer es comunicar la tarjeta con la computadora, para ello utilice el cable y conéctelo entre la tarjeta y un puerto serial de ella (COM1 o COM2).

Se debe utilizar el *software* de programación del microcontrolador, **figura 28.5**. Podemos cargar el siguiente programa y construir el circuito que aparece en la **figura 28.6**.

```
ciclo:
        low 0
        pause 500
        high 0
        pause 500
        goto ciclo
```

Una vez cargado el programa, el LED deberá encender en forma intermitente indicando salidas altas y bajas. Así comprobamos que la tarjeta ha quedado funcionando bien. En la sección de programación se muestran algunos ejercicios que pueden montarse en esta tarjeta.

Ensamblaje de la tarjeta de prototipos

En las **figuras 28.7** a la **28.12** se muestran los pasos para el ensamblaje de la tarjeta de prototipos para BASIC Stamp II, K-211.

Entrenador para Microcontroladores BASIC Stamp II

Los entrenadores son equipos que permiten hacer prácticas y experimentos de una forma rápida y sencilla. Contienen circuitos, o conjuntos de ellos, disponibles para que el usuario pueda, con unos pocos cables, montar, simular o probar el funcionamiento de un aparato o de un nuevo diseño.

Figura 28.5 *Entorno del software de programación del BASIC Stamp II*

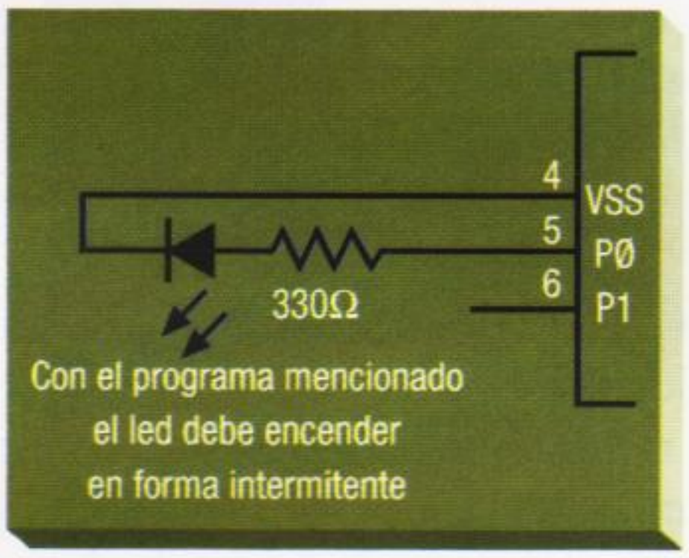


Figura 28.6 *Circuito de prueba de la tarjeta*

CEKIT pone a disposición de todos los interesados en esta tecnología un entrenador para el microcontrolador BASIC Stamp II, con el cual podremos llevar a cabo una serie de experimentos bastante interesantes.

Con este sistema de entrenamiento podremos hacer un sinnúmero de prácticas tales como el manejo de módulos LCD, comunicación serial RS-232, manejo de teclados y pantallas (*displays*), y muchas otras, ya que se incorpora en el equipo un *protoboard* con el que el experimentador puede crear sus propios ejercicios.

La tarjeta del entrenador está conformada por un conjunto de circuitos, independientes entre sí, para que el estudiante pueda hacer los más variados experimentos y proyectos sencillos (para los que se inician), o de cierta complejidad (para los más experimentados). Cada bloque tiene listas las conexiones a la fuente de alimentación y a los diferentes elementos que lo componen. De esta forma, solo se requiere conectar las entradas

Ensamblaje del proyecto

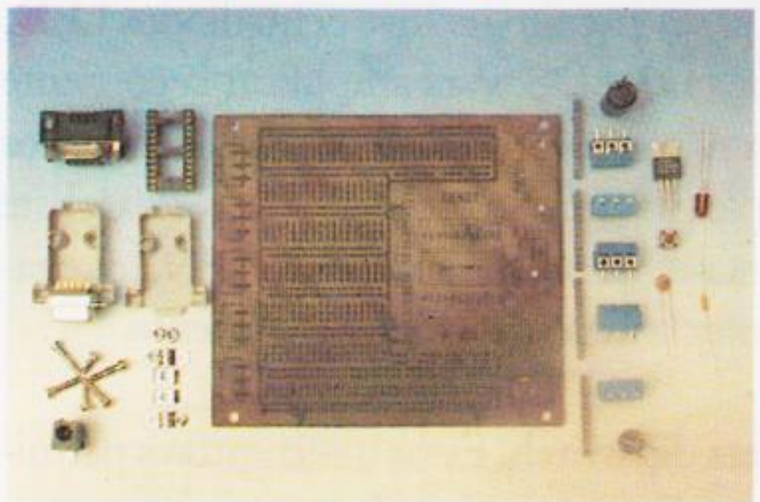
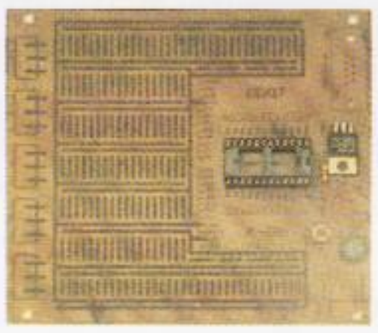


Figura 28.7 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar la tarjeta de prototipos del BASIC Stamp II, CEKIT K-211.*

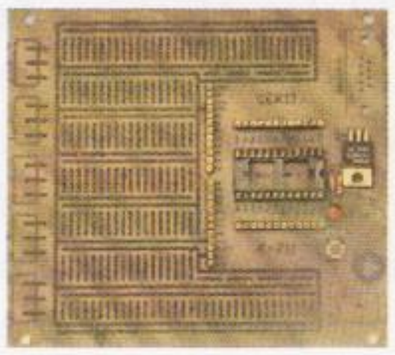
Esta tarjeta de prototipos se utiliza en el desarrollo, la programación, depuración y monitoreo de los circuitos de aplicación que se construyan con el microcontrolador BASIC Stamp II. Cuenta con una fuente regulada de 5V y 1A para alimentar cualquier circuito. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:



Paso 1

Instale y suelde la resistencia, la base para el microcontrolador, el pulsador de reset, el regulador de voltaje y el rectificador circular de 1A.

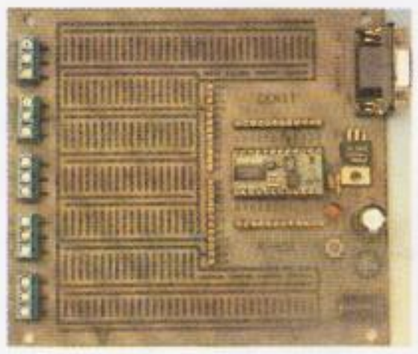
Figura 28.8



Paso 2

Luego instale y suelde el condensador cerámico, el diodo LED indicador de alimentación y los socket en línea que son una extensión para la conexión de todos los pines del microcontrolador.

Figura 28.9

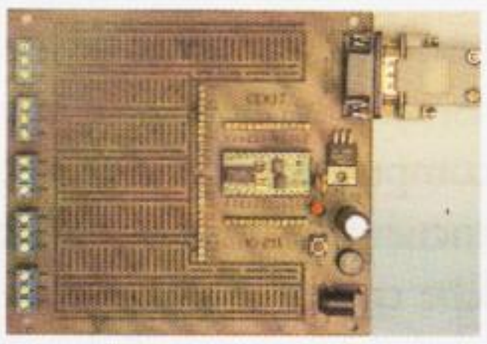


Paso 3

Instale y suelde los conextores de salida de tornillo y de propósito general, el conector serial DB-9 de programación, el conector de alimentación CA/CC y el condensador electrolítico de 1000µF.

Figura 28.10

LISTA DE MATERIALES		
Cant.	**Ref.**	**Descripción**
1	R1	330 Ω a 1/4W
1	C1	1000µF/35V electrolítico
1	C2	0.1µF/50V cerámico
1	PR1	Puente rectificador circular W06M
1	REG1	Regulador de voltaje integrado de +5V, 7805
1	D1	LED rojo de 5 mm
1	CN1	Conector DB9 hembra par circuito impreso
1	CN2	Conector CA/CC
1	SW1	Pulsador pequeño de cuatro pines para circuito impreso
1	K-211	Circuito impreso Cekit
1		Base para CI de 24 pines (ancha)
2		Espadines
5		Conectores de tornillo de 3 pines
4		Sockets en línea de 12 pines
1		Conector DB-9M con chasis

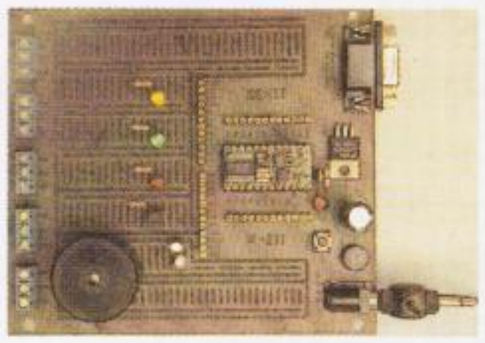


Paso 4

Aspecto final de la tarjeta de prototipo del BASIC Stamp II.

Figura 28.11

Ensamblaje de un ejercicio en la tarjeta de prototipos



Como un ejemplo para ilustrar el uso que se le puede dar a la tarjeta de prototipos, se ha construido en ella un circuito que usa un parlante piezoeléctrico, un pulsador y tres diodos LED para generar y monitorear la salida de tres señales seno con frecuencias diferentes. El diodo LED rojo indica la frecuencia de 2kHz, el verde la de 1kHz y el amarillo la de 500Hz; por medio del pulsador se cambia la frecuencia de salida.

Figura 28.12

Paso 5. Pruebas finales

Una vez ensamblada y asumiendo que todos sus componentes se encuentran en buen estado, es suficiente con conectar la fuente de alimentación dentro del rango indicado y el cable de programación que conecta la tarjeta con la PC. Por medio del diodo LED rojo encendido se indica el buen estado de la fuente regulada; y transfiriendo un programa cualquiera como uno de los mostrados en los ejercicos, se puede monitorear y disffrutar del buen funcionamiento de este sistema.

y/o las salidas al sitio que corresponda. Dichas conexiones se hacen fácilmente empleando alambre telefónico ya que se estableció un sistema de conectores que hacen esta labor muy sencilla. En la **figura 28.13** se muestra el diagrama esquemático de los circuitos que componen el entrenador. A continuación, describiremos cada uno de sus bloques.

Microcontrolador BASIC Stamp II. Como lo hemos visto en la sección de teoría, este microcontrolador tiene características de *hardware* y *software* especiales que lo hacen muy versátil y funcional. Entre las principales se encuentran: programación en lenguaje BASIC, instrucciones para comunicación serial, lectura de pulsadores, generación de tonos DTMF, 16 pines de entrada/salida, etc. El sistema entrenador pretende utilizar todas estas características, para lo cual se dispone de buses de expansión en los pines del microcontrolador. Recordemos que la conexión de la fuente de alimentación, el circuito de *reset* y el cable de programación, que se conecta a la computadora donde se escribe el programa, ya se encuentran instalados, por tanto, el usuario sólo debe conectar los puertos o pines de entrada y salida según su necesidad.

Fuente de alimentación. El sistema se alimenta de la línea CA de 110 ó 220V, incluye un interruptor general y un fusible de protección. El entrenador incluye una fuente interna con tensiones de salida de +5 y ±12 VCC. La fuente de 5V se utiliza para alimentar la parte digital del sistema y la fuente dual de ±12V sirve, por ejemplo, para alimentar amplificadores operacionales que se monten en el *protoboard* del entrenador.

Diodos LED. El circuito incluye ocho diodos LED, denominados D1 a D8. Para amplificar la corriente que entrega el microcontrolador para encender los LED se emplea un circuito integrado ULN2803. Éste posee internamente ocho transistores NPN, los cuales tienen muy buena capacidad de corriente.

Interruptores dipswitch. Estos interruptores permiten que el estudiante practique la lectura de puertos. En el circuito se emplea un *dipswitch* de 4 interruptores, denominados S1 a S4. Cuando un interruptor se encuentra apagado (posición OFF), el microcontrolador leerá un nivel lógico alto (+5V), cuando el interruptor se encuentra activado (posición ON), la lectura entregará un nivel lógico bajo (0V).

Pulsadores. En el sistema se incluyen dos pulsadores normalmente abiertos, los cuales también son de suma importancia cuando se ejercita la lectura de puertos. Se encuentran referenciados como S5 y S6. En estado de reposo, la lectura de estos pines entrega un nivel lógico alto (+5V), cuando se presionan, la lectura será un nivel lógico bajo (0V).

Interfaz RS-232. La comunicación serial RS-232 es una de las más utilizadas en la industria. Para establecerla, sólo se requiere de un circuito integrado MAX232 y cinco condensadores de 10µF (tantalio). Los pines de comunicación se denominan Tx y Rx. El cable que une el entrenador y la computadora o equipo con el que se quiere establecer comunicación se hace uno a uno (sin cruces).

Salida con relé. Para manejar cargas de potencia

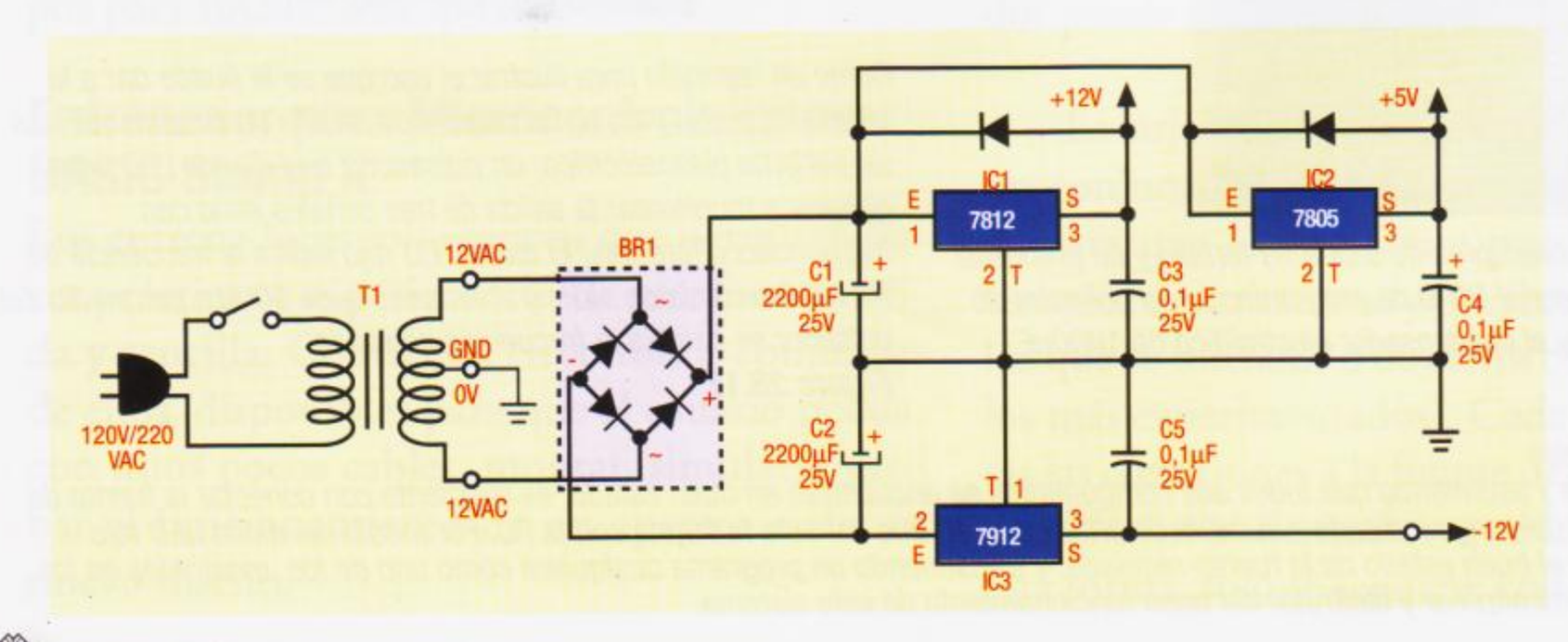


Figura 28.13a
Fuente de alimentación

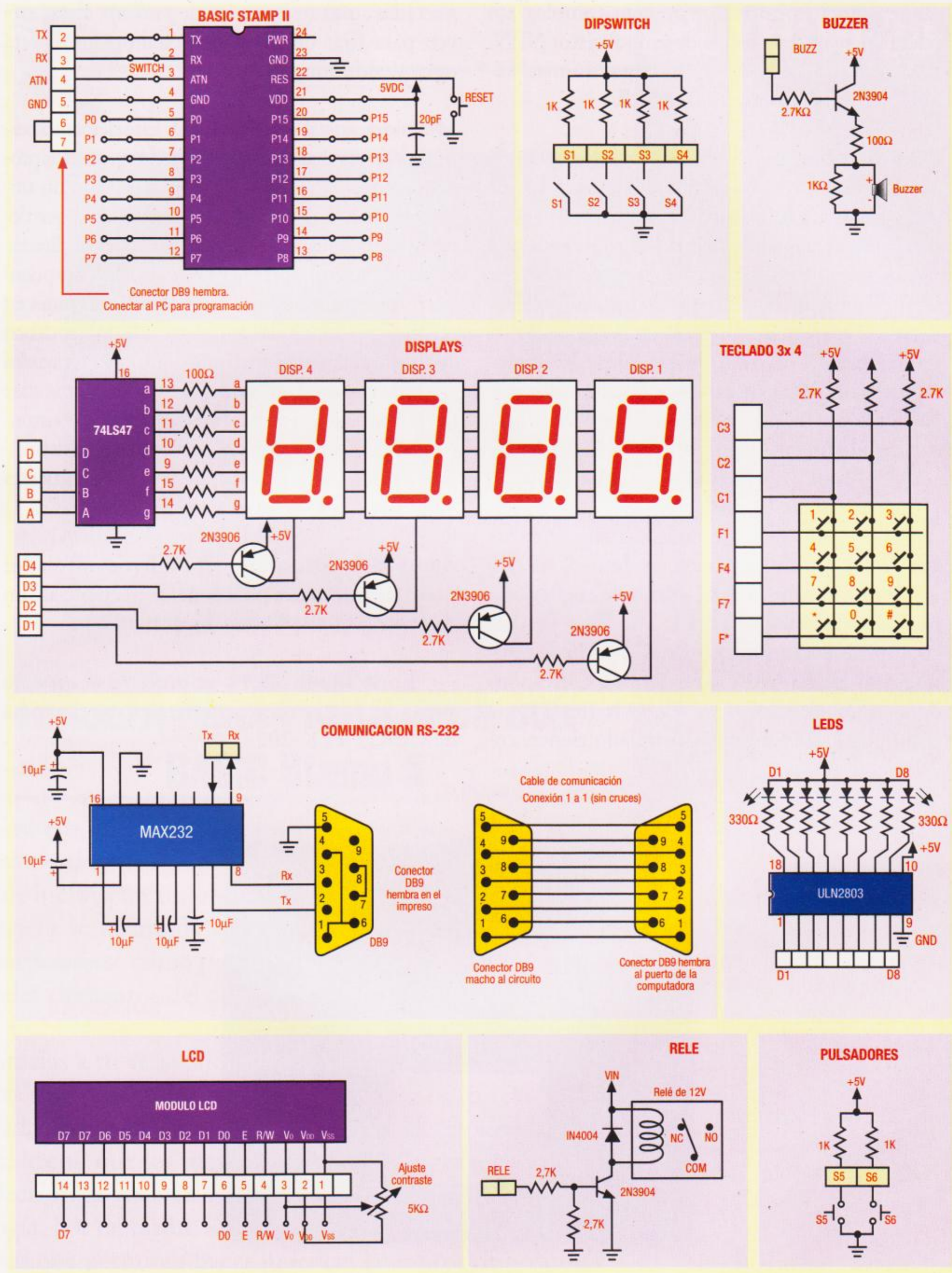


Figura 28.13b Circuitos que componen el entrenador

desde el microcontrolador, podemos emplear un relé. Éste se opera a través de un transistor NPN. Para activarlo, sólo se debe entregar un nivel lógico alto en el pin marcado RELE.

Displays. El sistema dispone de cuatro *displays* de ánodo común, conectados sobre el mismo bus de datos y con los ánodos manejados de forma independiente. Para la conversión del dato que entrega el microcontrolador al código 7 segmentos, se emplea un decodificador 7447. Todos los segmentos de los *displays* se encuentran conectados a los pines de salida del 7447. Por lo tanto, el dígito que se desea mostrar en ese momento se debe seleccionar mediante la habilitación del *display* correspondiente. Esto se hace conectando un nivel lógico bajo en la base del transistor PNP que maneja el ánodo de dicho *display*.

Teclado matricial. Este teclado tiene una matriz de 3 x 4, con los caracteres desde 0 hasta 9 y * y #. En el conector del mismo se encuentran marcadas las filas como F1, F4, F7 y F*. Las columnas se denominan C1, C2 y C3. La numeración de las filas y las columnas corresponde al primer carácter, ya sea de la fila o de la columna. Las columnas del teclado tienen conectadas unas resistencias de *pull-up*. Éstas sirven para fijar un nivel alto cuando no se está oprimiendo ninguna tecla.

Módulo de cristal líquido o LCD. Este elemento permite darle mucha funcionalidad y elegancia a los proyectos; además, es muy fácil de manejar con un microcontrolador. Si se utiliza un módulo LCD que tiene todos los pines en línea, se puede conectar directamente al circuito impreso sin necesidad de hacer puentes. Si, por el contrario, el módulo tiene los pines en un conjunto de dos filas de 7 pines cada uno, se deben conectar los puentes de alambre que están marcados en el circuito impreso. Normalmente esta última clase tiene además luz posterior (*backlight*), se debe entonces conectar los dos pines del extremo derecho del módulo y la resistencia de 10Ω (1/2W) que se encuentra junto al potenciómetro de ajuste del brillo.

Zumbador. Dado que este microcontrolador posee instrucciones para generar frecuencias, un zumbador (*buzzer*) es de de gran utilidad.

En la **figura 28.14** se muestra el aspecto físico del entrenador completo cuya referencia de CEKIT es K-202.

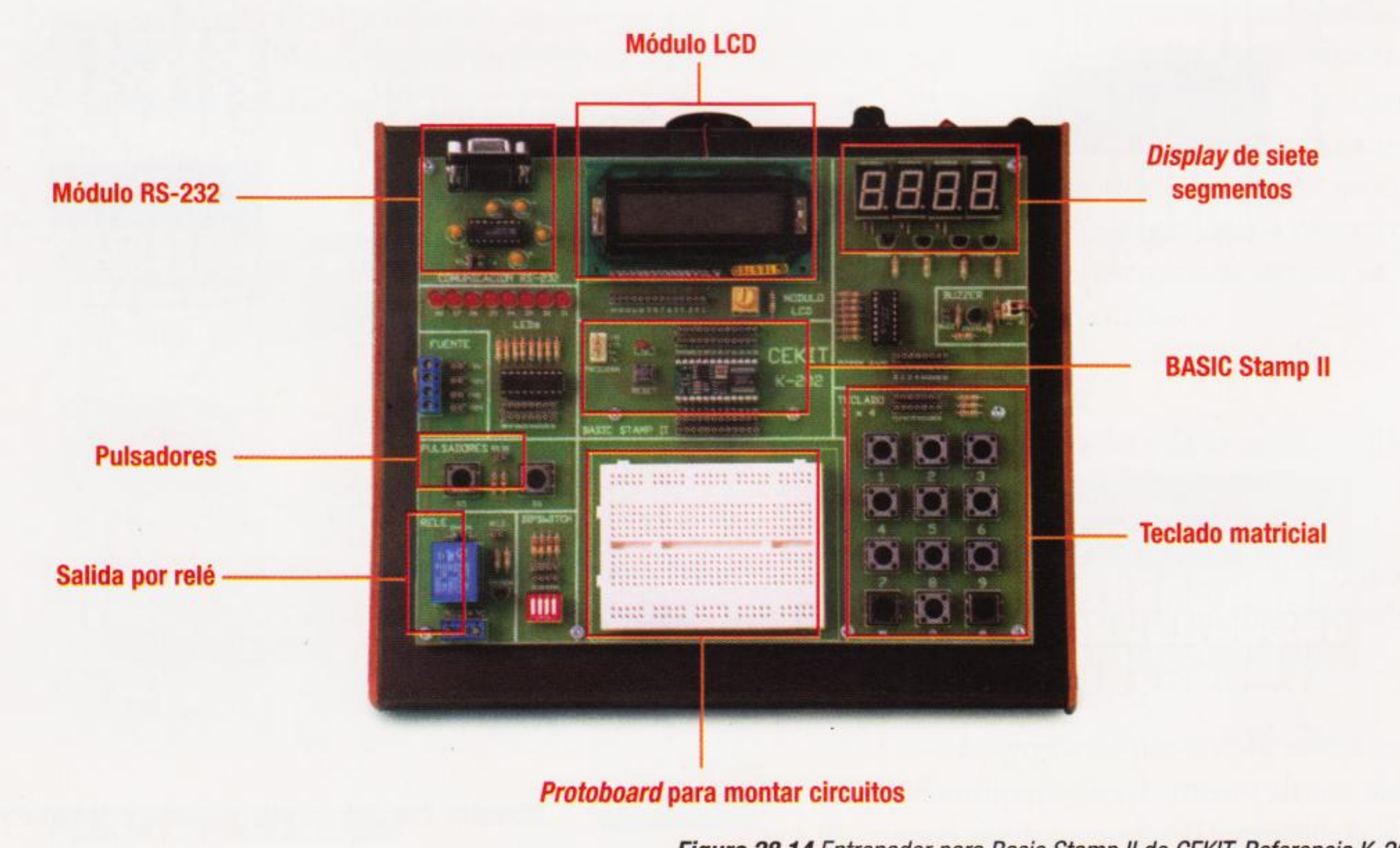
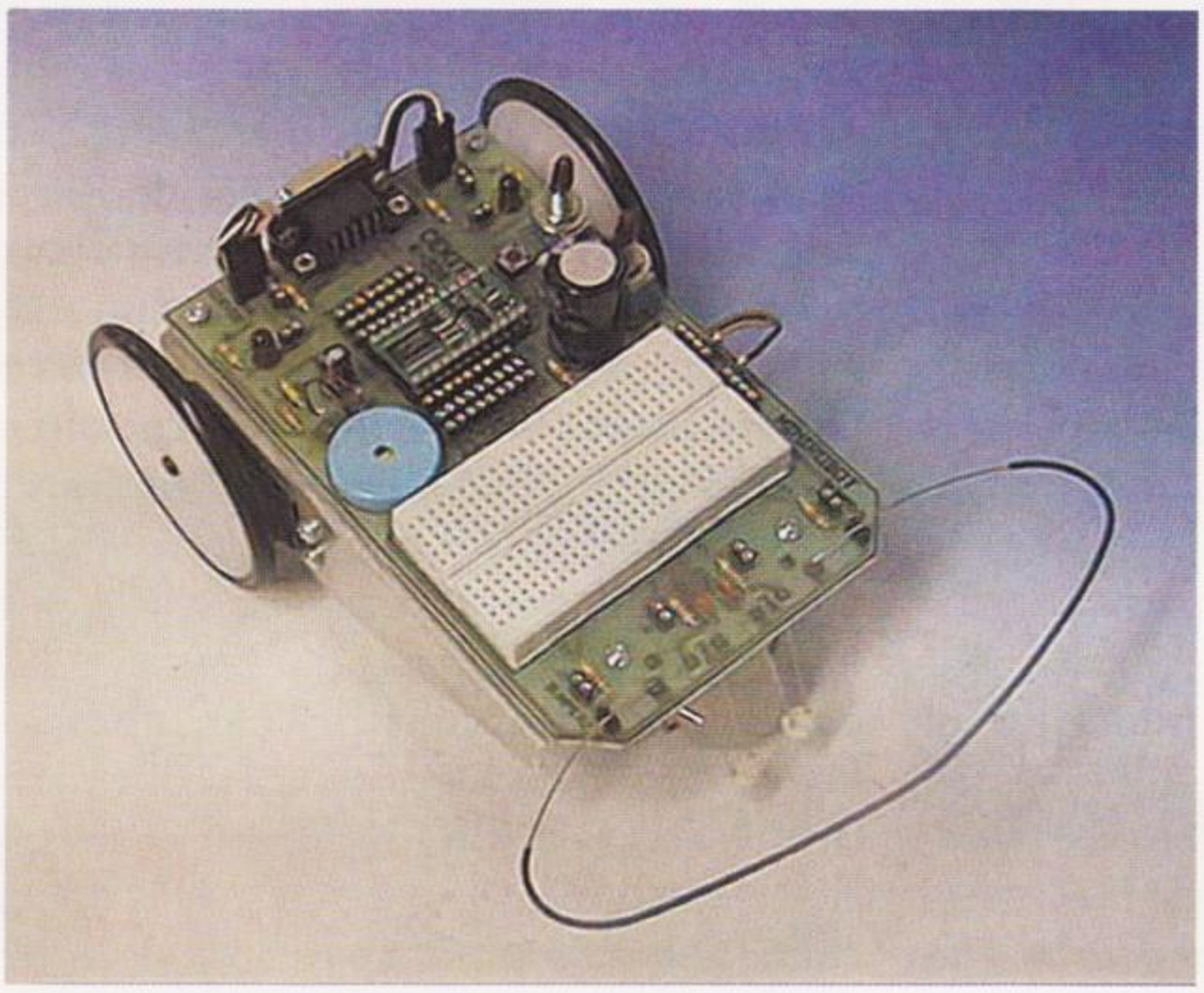


Figura 28.14 *Entrenador para Basic Stamp II de CEKIT. Referencia K-202*

PROYECTO 30



MINIROBOT CON EL BASIC STAMP II

Este proyecto es ideal para iniciarse en la robótica debido a su fácil construcción y modo de programación, al utilizar el lenguaje PBASIC de los microcontroladores Basic Stamp. La estructura del robot es de tipo plataforma y su movimiento lo proporcionan dos servomotores similares a los utilizados en aeromodelismo y vehículos pequeños de control remoto.

Características

- Tracción con dos ruedas impulsadas por servomotores
- Control con un BasicStamp II
- Sensores: fotoceldas y parachoques (*bumper*) electromecánico
- Indicadores: dos diodos LED y un zumbador piezoeléctrico
- Conexiones configurables
- Programación con una PC por puerto serial RS232
- Alimentación con cuatro pilas alcalinas o recargables AA de 1,5 V
- *Protoboard* para circuitos adicionales

En las prácticas de un curso de microcontroladores no puede faltar el ensamblaje y programación de un robot. En esta oportunidad estamos presentando una plataforma básica controlada por un BasicStamp II la cual permite realizar una buena cantidad de ejercicios con el fin de familiarizarnos e ir aprendiendo sobre esta tecnología.

Estructuras eléctrica y mecánica del robot

Como la gran mayoría de los robots, hay una parte electrónica y otra mecánica. Esta última es del tipo plataforma con dos ruedas impulsadas cada una por un motor independiente y una rueda libre para darle estabilidad. Este robot, con referencia K-338 de CEKIT, está basado en algunos modelos de Parallax, fabricante del BASIC Stamp, pero tiene sus características propias, principalmente la posibilidad de configurar sus conexiones en diferentes formas. Los dos motores, que se aseguran en los lados de la plataforma, son del tipo servomotor o "servo", a los cuales se les hace una modificación con el fin de que puedan girar los 360° que se requieren para el movimiento continuo de las ruedas.

La plataforma, fabricada en aluminio, lleva en su parte superior el circuito impreso que contiene todos los elementos electrónicos del sistema incluyendo el control principal que es un microcontrolador BASIC Stamp II (BS2-IC), el cual ha sido ampliamente explicado en las secciones de teoría y programación de este curso. El diagrama esquemático del circuito electrónico se muestra en la **figura 30.1**. La fuente de alimentación es un conjunto de cuatro pilas de 1,5 voltios para un total de 6 voltios que se utilizan para alimentar los motores y el microcontrolador; el soporte para las pilas está ubicado por debajo de la plataforma.

Se incluye un conector tipo DB-9 hembra por medio del cual se programa el microcontrolador desde una PC. En el kit se incluye un disquete con algunos programas básicos de prueba. A cada lado del microcontrolador, en los pines de los puertos de entrada y salida (I/O), hay tres conectores con todos sus pines disponibles que permiten conectar los elementos externos, como los motores, los sensores, los indicadores etc. para diferentes tareas y poder hacer la expansión del sistema. Además se tiene disponible un *protoboard* para montar en él diferentes tipos de circuitos experimentales con otros tipos de sensores, módulos de RF para comunicaciones inalámbricas, etc.

Como principal sensor del robot se tiene un parachoques (*bumper*) formado con una pieza de alambre acerado que hace contacto entre dos conectores, lo que cambia el nivel lógico de un pin (1 ó 0) cuando el robot toca un obstáculo. También tiene dos fotorresistencias o "fotoceldas" que se pueden utilizar para detectar diferentes niveles de iluminación y seguir una ruta trazada con una línea blanca en el piso, por ejemplo, u orientarse por medio de la iluminación de un recinto. Como elementos de entrada y salida, el robot tiene dos diodos LED indicadores y un interruptor tipo pulsador para darle "*reset*" o reiniciar el circuito y un pequeño transductor sonoro (parlante piezoeléctrico miniatura). Estos elementos se pueden conectar en forma individual para que el usuario los utilice cuando sea necesario.

Los servomotores

Estos son un tipo especial de motores, **figura 30.2**, que solamente giran una porción de la circunferencia, generalmente entre 90° y 180°. Debido a la facilidad para controlar el ángulo de

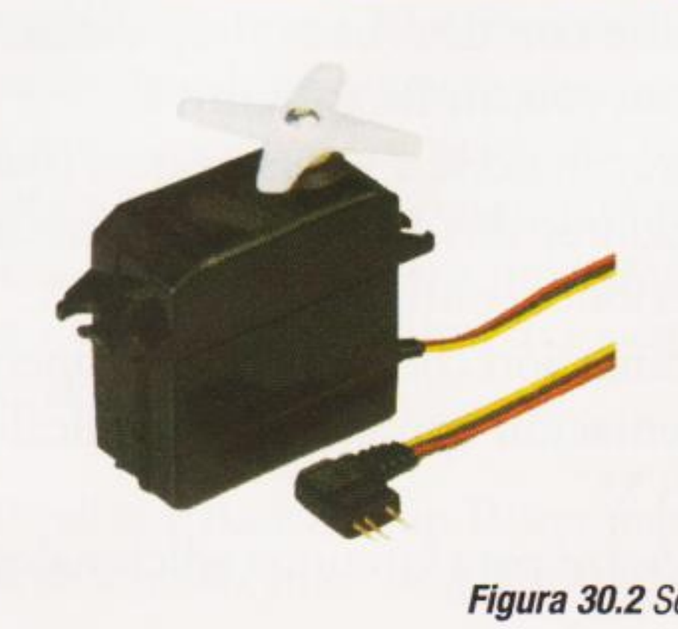


Figura 30.2 *Servomotores*

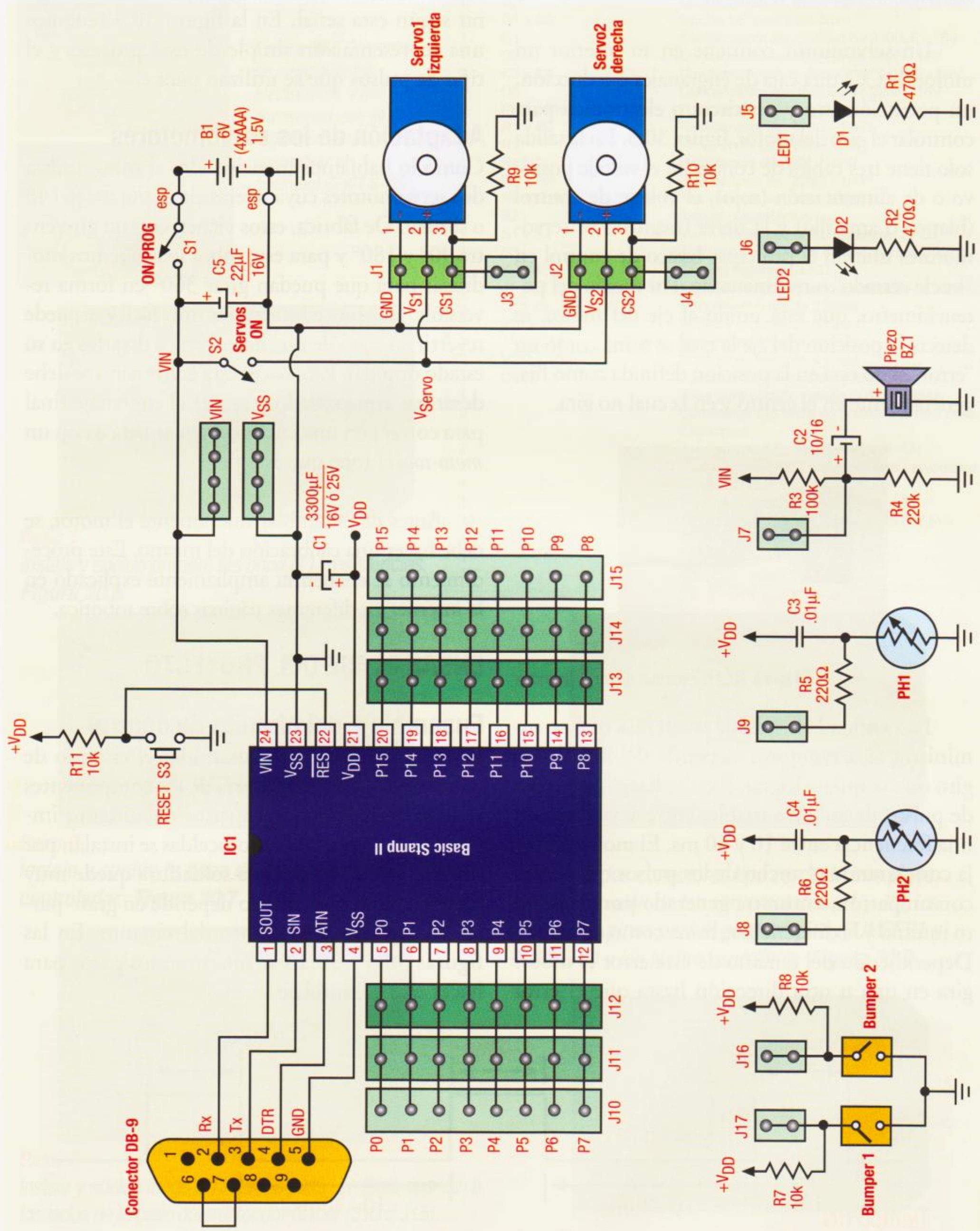


Figura 30.1 Diagrama esuqemático del robot con microcontrolador BasicStamp II

giro y a su relativo bajo costo, son muy empleados en robótica experimental.

Un servomotor contiene en su interior un motor de CC, una caja de engranajes o reducción, un potenciómetro y un circuito electrónico para controlar el giro del motor, **figura 30.3.** En la salida solo tiene tres cables de conexión: el voltaje positivo o de alimentación (rojo), el voltaje de control (blanco o amarillo) y la tierra (negro). Los servomotores utilizan el principio básico de control en "bucle cerrado" o realimentado. Por medio del potenciómetro, que está unido al eje del motor, se detecta la posición del eje la cual se toma como un "error" si no está en la posición definida como fija, generalmente en el centro y en la cual no gira.

Figura 30.3 *Elementos de un servomotor*

La cantidad de voltaje o energía que se suministra al servomotor depende del ángulo de giro que se quiera lograr. Esto se hace por medio de pulsos de ancho variable, entre 1 y 2 ms con una frecuencia entre 10 y 30 ms. El motor trabaja comparando el ancho de los pulsos que recibe con un patrón o muestra generado por el circuito interno y la diferencia se toma como el "error". Dependiendo del tamaño de este error el motor gira en una u otra dirección hasta que el error desaparezca, momento en el cual se detiene. Si cambia la señal de control, el motor vuelve a girar según esta señal. En la **figura 30.4** tenemos una representación simple de este proceso y el tipo de pulsos que se utilizan para él.

Adaptación de los servomotores

Como lo habíamos mencionado, el robot utiliza dos servomotores cuya referencia es Futaba S-148 o similar. De fábrica, estos vienen con un giro entre 90° y 180° y para este robot los debemos modificar para que puedan girar 360° en forma reversible. Este procedimiento es muy fácil y se puede revertir en caso de que necesitemos dejarlos en su estado original. Para hacer esta conversión se debe desarmar con cuidado y retirar el engranaje final para cortar con una cuchilla bien afilada o con un *moto-tool* el tope que éste tiene.

Antes de ensamblar nuevamente el motor, se debe hacer una calibración del mismo. Este procedimiento se encuentra ampliamente explicado en la internet en diferentes páginas sobre robótica.

ENSAMBLAJE DEL PROYECTO

Ensamblaje del circuito de control

Inicialmente debemos ensamblar el circuito de control en el cual la mayoría de los componentes se instalan en la parte superior del circuito impreso, solamente las dos fotoceldas se instalan por debajo. Observe que cada soldadura quede muy bien hecha ya que de ello depende en gran parte el buen funcionamiento del circuito. En las **figuras 30.5** a **30.13** se muestran los pasos para hacer este ensamblaje.

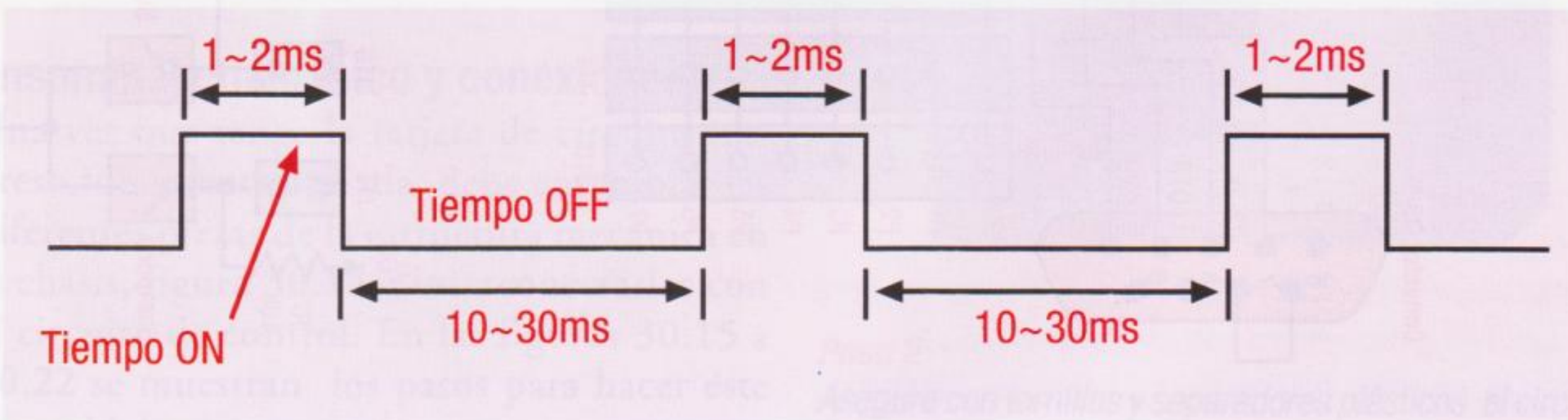


Figura 30.4 *Señal de pulsos que se debe aplicar a un servomotor*

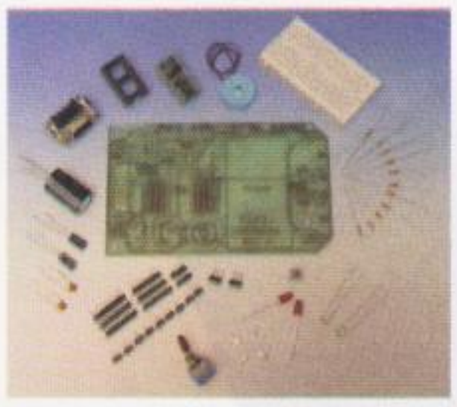
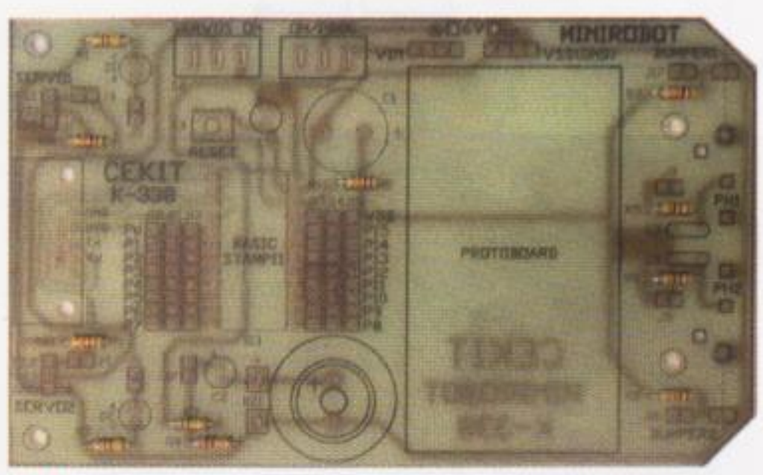


Figura 30.5 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el minirobot CEKIT K-338.*

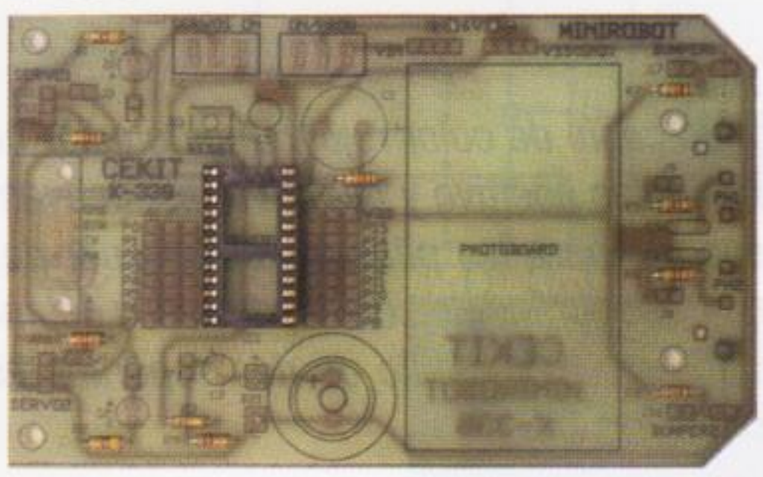
LISTA DE MATERIALES

Cant.	Ref.	Descripción
1	IC1	Microcontrolador Basic Stamp II, programado
2	D1 y D2	Diodos LED rojos de 5mm
1	C1	Condensador electrolítico de 3300uF - 16V
1	C2	Condensador electrolítico de 10uF - 16V
2	C3 y C4	Condensadores cerámicos de 0,01uF - 50V
1	C5	Condensador electrolítico de 22uF - 16V
2	PH1 y PH2	Fotoceldas
2	R1 y R2	Resistencias de 470Ω - 1/4W
1	R3	Resistencia de 100k - 1/4W
1	R4	Resistencia de 220k - 1/4W
2	R5 y R6	Resistencias de 220Ω - 1/4W
5	R7 a R11	Resistencias de 10k - 1/4W
1	BZ1	Buzzer piezoeléctrico
2	S1 y S2	Interruptores de codillo miniatura, SPDT
1	S3	Pulsador pequeño de 4 pines para impreso
1	K-338	Circuito impreso CEKIT
2	J1 y J2	Conectores tipo cerca no polarizados de 3 pines
9	J3 a J9, J16 y J17	Sockets en línea de 2 pines
3	J10 a J12	Sockets en línea de 8 pines
3	J13 a J15	Sockets en línea de 9 pines
2	VDD y VSS	Sockets en línea de 4 pines
1	DB9F	Conector DB9F para circuito impreso
1		Base para circuito integrado de 24 pines
2		Conectores tipo cerca no polarizados de 2 pines
4		Espadines
2		Servomotores Futaba S-148 o similar
2		Ruedas de plástico y caucho para servomotor
1		Chasis de aluminio CEKIT K-338
1		Soporte para cuatro pilas AA de 1,5V
12		Tornillos con tuerca hexagonal de 3x7 mm
1		*Protoboard* pequeño
1		Parachoques de alambre acerado
1		Juego de cables de conexión



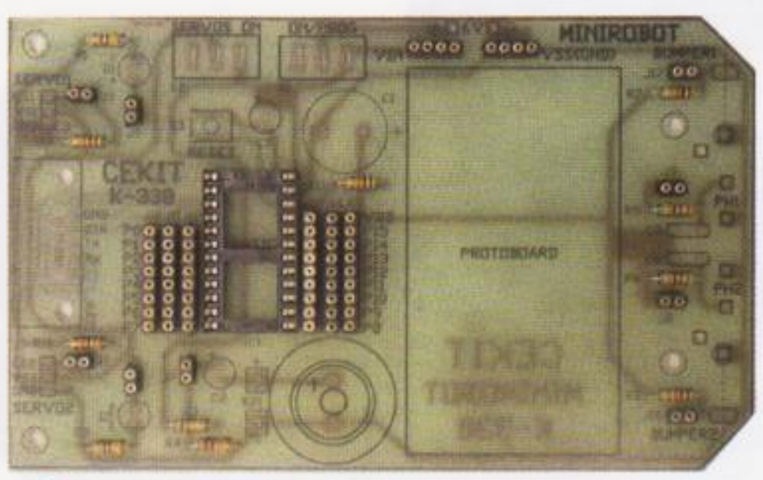
Paso 1

Instale y suelde primero las once (11) resistencias. ***Figura 30.6***



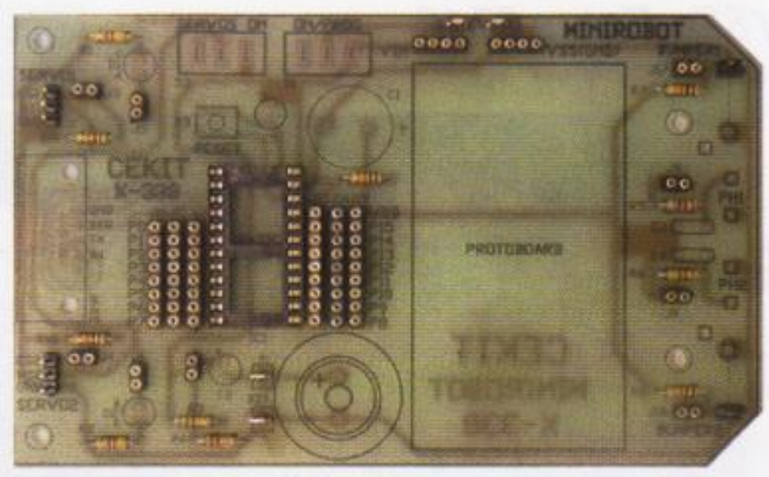
Paso 2

Instale y suelde la base de 24 pines para el microcontrolador. ***Figura 30.7***



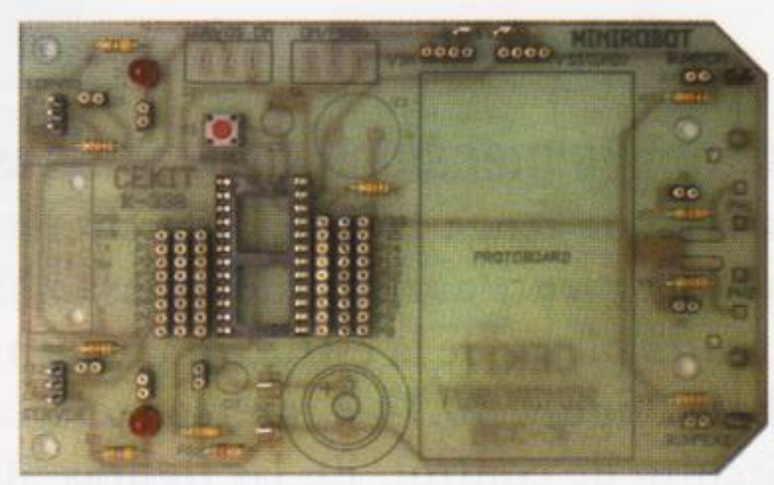
Paso 3

Instale y suelde luego los seis "sockets" en línea que van a los lados de la base del microcontrolador (J10 a J15). Luego los de cuatro pines (VIN y Vss). Después monte los nueve sockets en línea de dos pines (J3, J4, J5, J6, J7, J8, J9, J16 y J17). ***Figura 30.8***



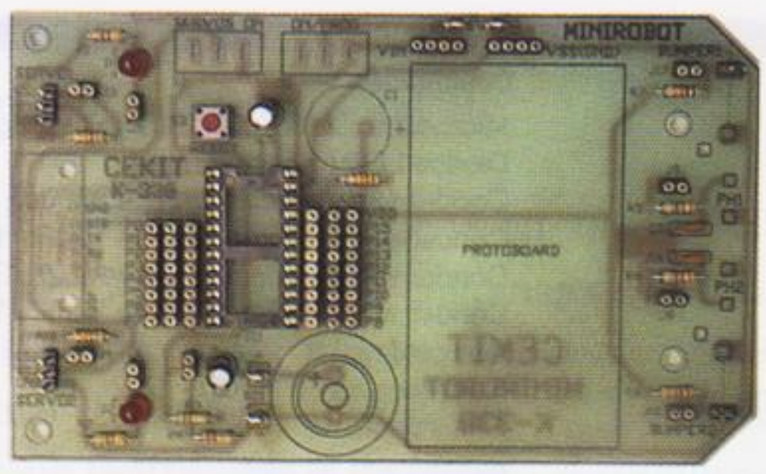
Paso 4

Instale los dos conectores tipo cerca de tres pines para la conexión de los servomotores (J1 y J2), los dos de dos pines para el parachoques (BUMPER1 y BUMPER2) y los cuatro espadines. ***Figura 30.9***

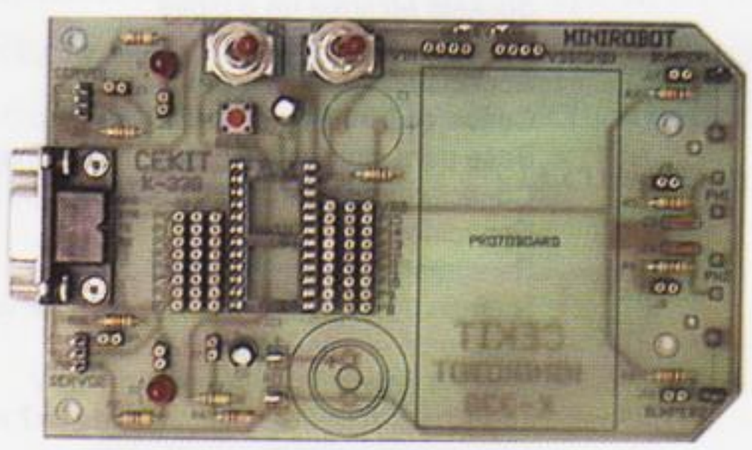


Paso 5

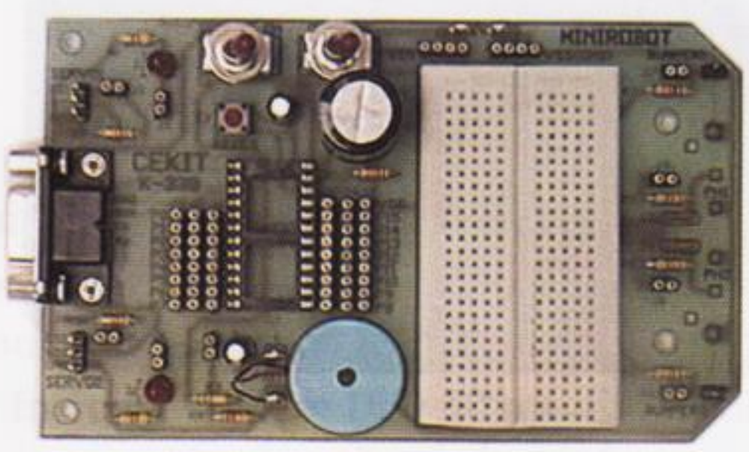
Instale los diodos LED y el pulsador S3. ***Figura 30.10***



Paso 6
Instale los condensadores cerámicos (C3 y C4), los condensadores electrolíticos pequeños (C2 y C5). ***Figura 30.11***



Paso 7
Instale el conector DB-9 y los interruptores de codillo (S1 y S2). ***Figura 30.12***

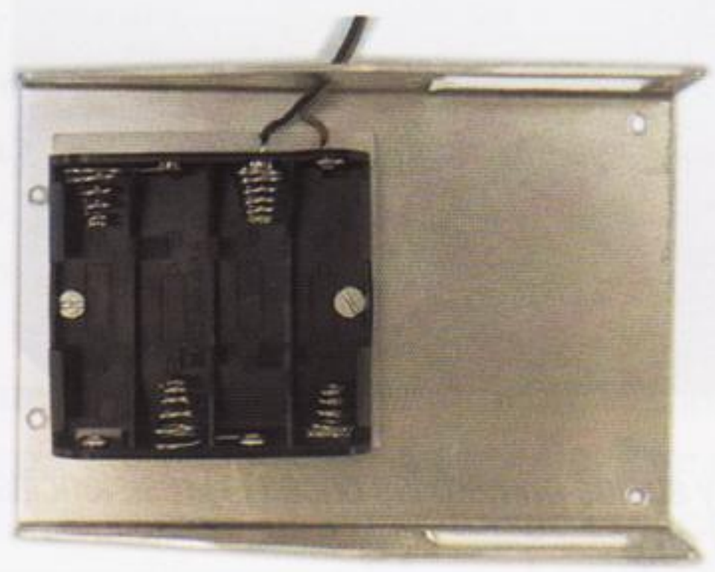


Paso 8
Instale el protoboard, el parlante piezoeléctrico o buzzer y el condensador electrolítico grande (C1). ***Figura 30.13***

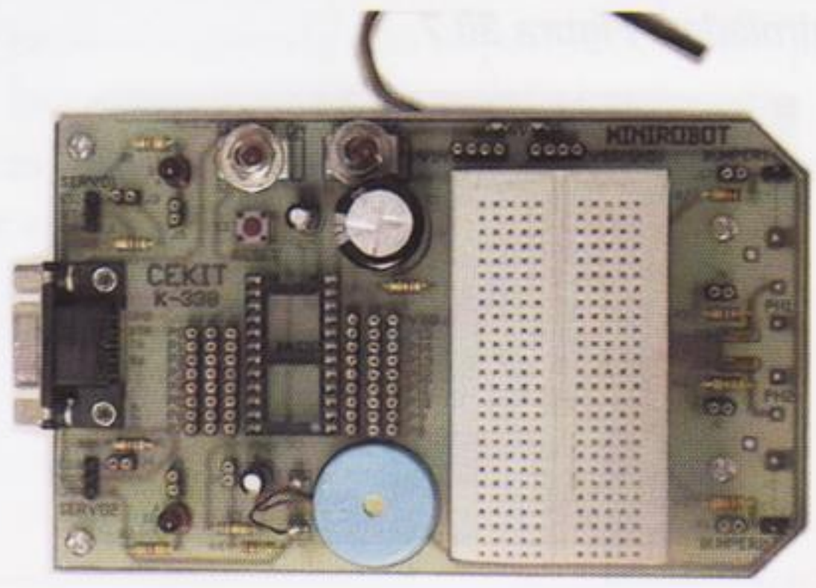
Ensamblaje mecánico y conexiones

Una vez que tenga la tarjeta de circuito impreso totalmente armada, debe ensamblar las diferentes piezas de la estructura mecánica en el chasis, **figura 30.14** e interconectarlas con el circuito de control. En las **figuras 30.15** a **30.22** se muestran los pasos para hacer este ensamblaje.

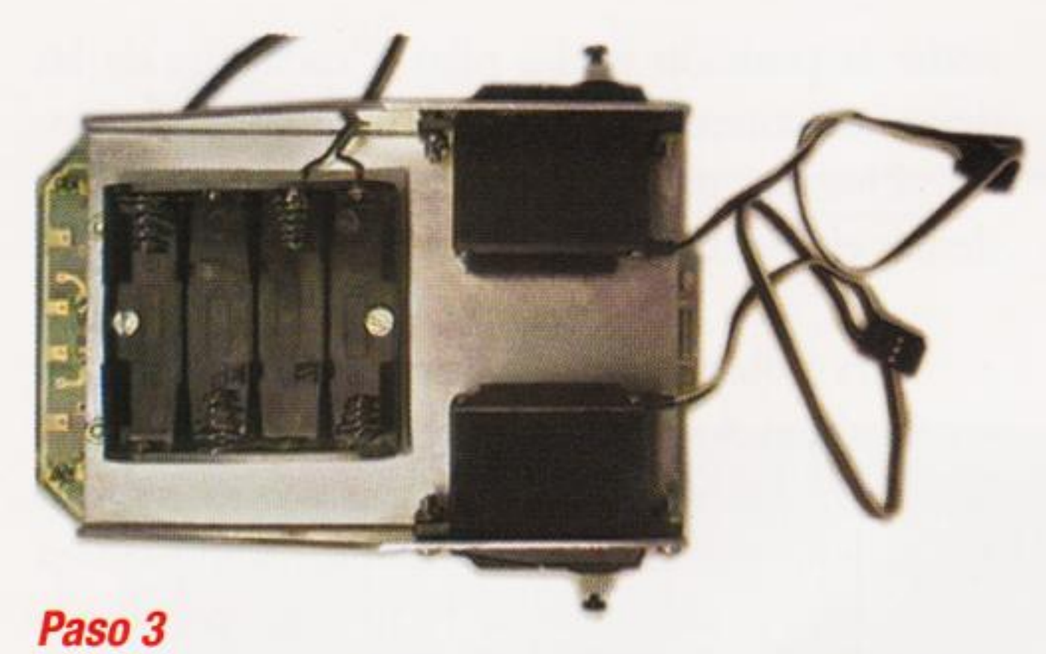
Figura 30.14 *Chasis de aluminio para el montaje del robot*



Paso 1
Conecte un cable de color rojo y uno de color negro a los terminales positivo y negativo respectivamente del soporte para las pilas e instálelo por debajo del chasis, asegurándolo con dos tornillos dejándolo aislado de éste, por medio de dos separadores de plástico. ***Figura 30.15***

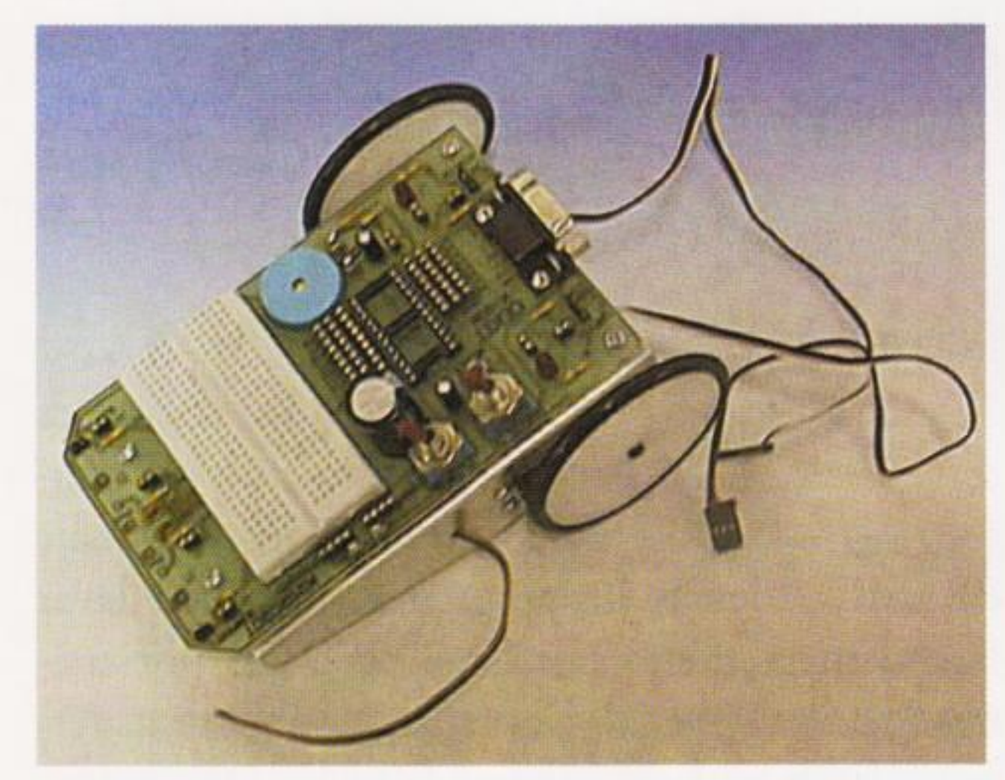


Paso 2
Asegure con tornillos y separadores plásticos el circuito impreso en el chasis del robot. ***Figura 30.16***



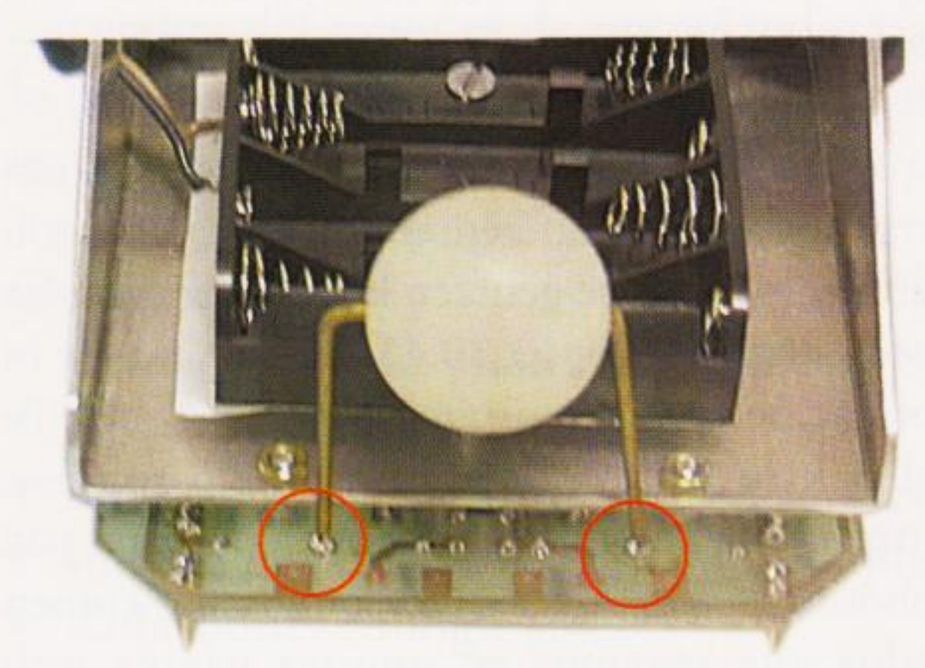
Paso 3

Asegure los dos servomotores con cuatro tornillos, cada uno en la parte inferior del chasis con los ejes hacia la parte trasera del robot. ***Figura 30.17***



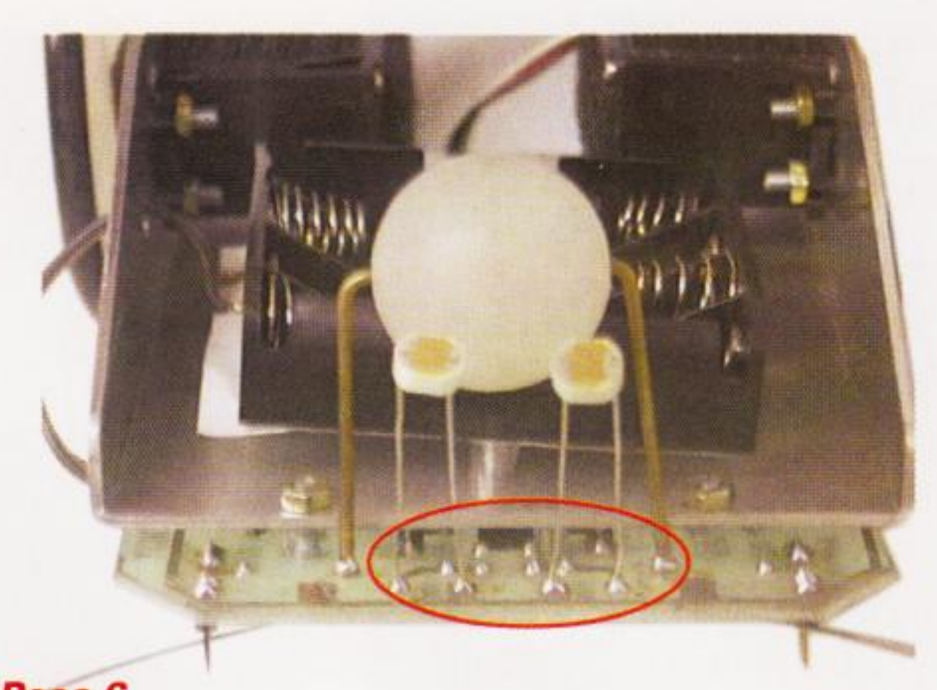
Paso 4

Instale las ruedas en los ejes del motor y asegúrelas con los tornillos adecuados. ***Figura 30.18***



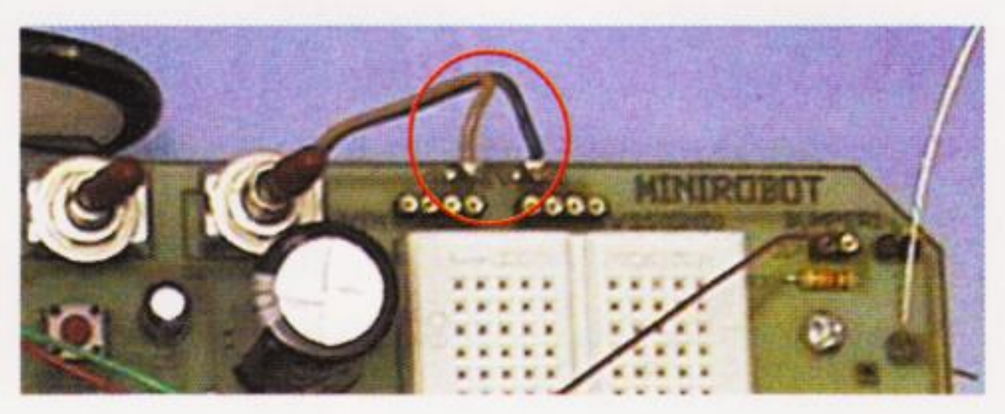
Paso 5

Instale y suelde el soporte delantero de alambre con la rueda guía en los orificios destinados para tal fin en el circuito impreso. ***Figura 30.19***



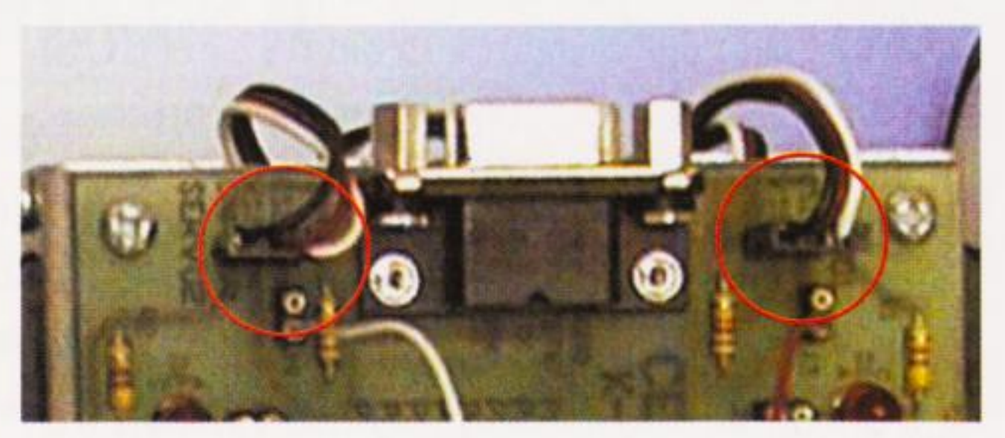
Paso 6

Instale y suelde las dos fotoceldas (PH1 y PH2) en la parte delantera. ***Figura 30.20***



Paso 7

Luego conecte el cable de las pilas en los terminales positivo y negativo de alimentación del robot (+ 6V -), conservando la polaridad adecuada. ***Figura 30.21***



Paso 8

Después instale los terminales de los servomotores en los conectores J1 y J2. El alambre negro debe ir en el punto marcado como GND. ***Figura 30.22***

Paso 9

Para asegurarnos que la alimentación del microcontrolador esté correcta, mida el voltaje entre los pines 24 y 23 de su base antes de instalarlo. Debe encontrar un valor cercano a los + 6V si las pilas están nuevas. Instale el microcontrolador BasicStamp en su base observando bien su posición. El pin No.1 debe quedar al lado del interruptor de RESET.

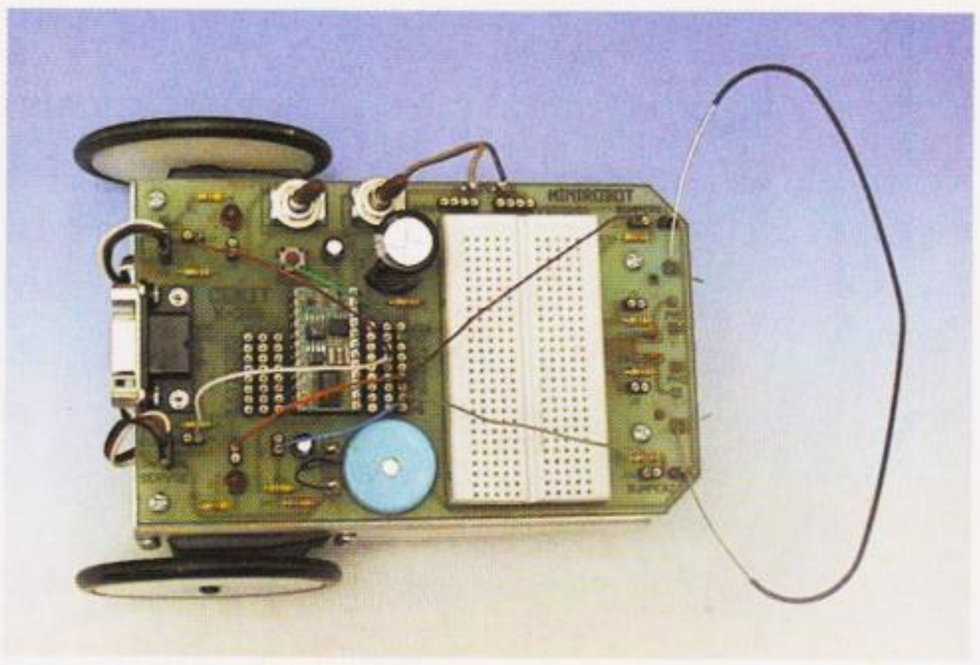


Figura 30.23 *Conexión de los elementos externos al microcontrolador*

Interconexión de los elementos externos a la tarjeta de control

Para que el robot funcione, debemos conectar los diferentes elementos (motores, diodos LED, parlante y *bumper*) a los respectivos puertos del microcontrolador haciendo puentes de alambre a través de los diferentes conectores. Antes de hacerlo, verifique que el interruptor general S1 esté apagado (hacia el *protoboard*). Haga las conexiones así: el motor izquierdo (J3) al pin P13 del BS2, el motor derecho (J4) al pin P12 del BS2, el LED izquierdo (J5) al pin P15 del BS2, el LED derecho (J6) al pin P11 del BS2, el zumbador (J7) al pin P8 del BS2, el BUMPER1 (J17) al pin P10 del BS2 y el BUMPER2 (J18) al pin P9 del BS2. Por ahora no conecte las fotoceldas. **Figura 30.23**

Prueba inicial de la tarjeta

Puede hacer una prueba preliminar de la tarjeta de control por medio del programa ***robotest.bs2,*** el cual se puede descargar de la página web de CEKIT. Antes de hacerla debe llevar el programa a la memoria del microcontrolador, tal como lo explicamos en la sección de programación de este curso. Para hacer la prueba, lleve el interruptor de los servomotores a su posición de apagado (hacia S1) e instale las pilas en su soporte. Lleve ahora el interruptor general S1 a su posición de encendido (ON), los diodos LED se deben encender en forma intermitente y se debe escuchar también un sonido intermitente en el zumbador. Si esto ocurre, se confirma que buena perte del circuito está funcionando. Si las pruebas no funcionan, revise la posición de las pilas y cada una de las diferentes conexiones. Después de corregir los errores, si los hay, pasamos a probar los motores y los demás elementos para terminar el robot.

Para probar el robot con sus motores y el *bumper*, sosténgalo en el aire y active el interruptor S2 (hacia el LED D1). En este momento las ruedas deben girar. Accione cada lado del *bumper* para que el alambre haga contacto con cada uno de los terminales por donde pasa y los motores deben cambiar de dirección en los diferentes casos (accionamiento izquierdo y derecho). Si no es así, hay un problema en el programa o en las conexiones de los servomotores, revíselos bien y corrija los errores de *software* o los de las conexiones hasta que el robot opere correctamente, funcionando en forma autónoma en el piso y cambiando de rumbo cuando encuentra un obstáculo.

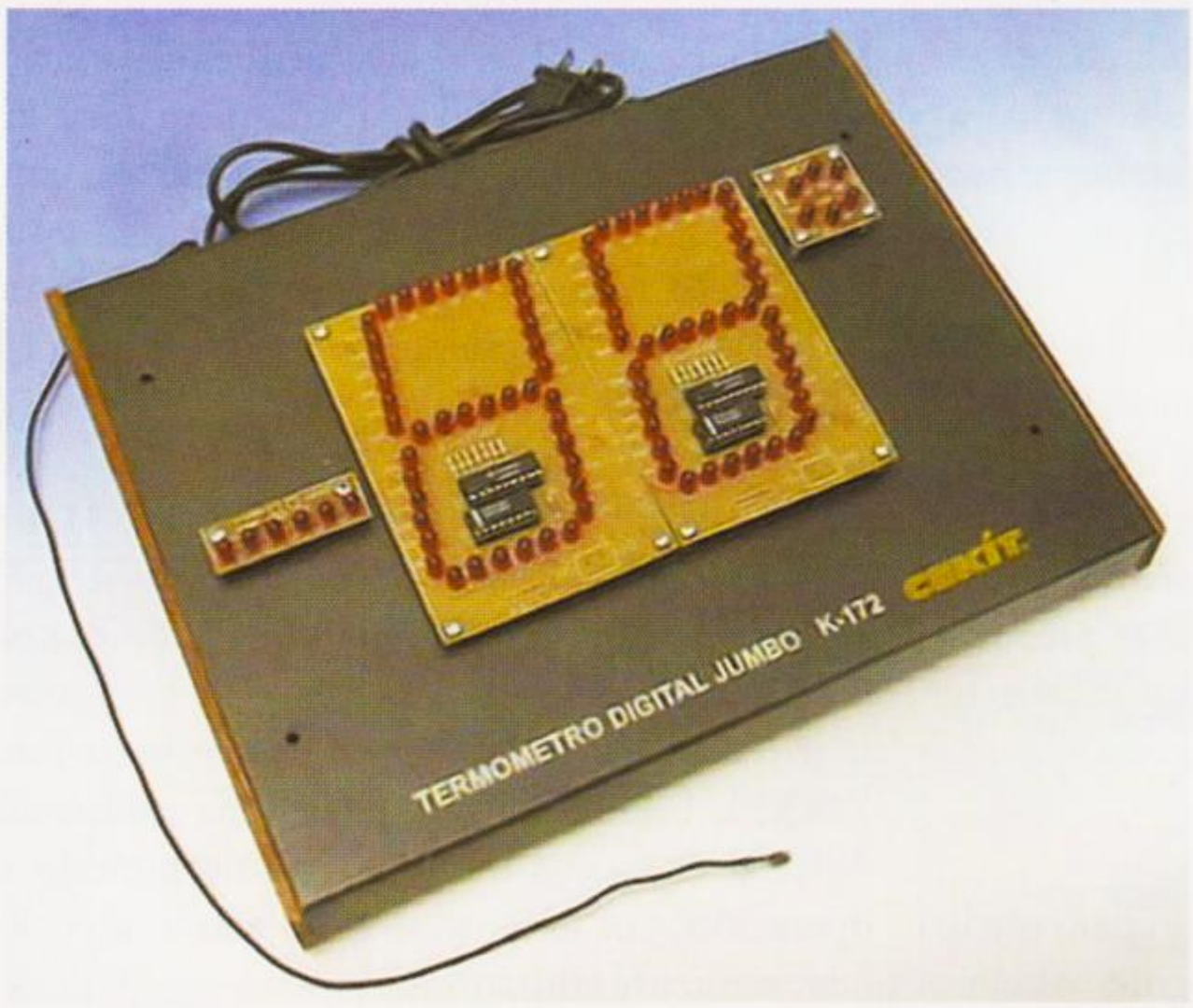
Programación del robot

Una de las grandes ventajas de este robot es que es totalmente configurable en cuanto a sus elementos externos y programable utilizando el lenguaje PBASIC. Además, los programas se pueden llevar fácilmente al microcontrolador BS2-IC por medio de un cable serial. Configure el robot de acuerdo a sus ideas y necesidades utilizando esta estructura robótica básica. Utilizando el *protoboard*, se pueden agregar otros circuitos con diferentes tipos de sensores, un receptor de infrarrojos, un receptor y transmisor de RF, etc., con el fin de brindarle nuevas funciones a este práctico robot.

Para el manejo de los servomotores y su interacción con los diferentes sensores, debe utilizar algoritmos que generen las señales de pulsos mostradas en la figura 30.4. Para una mayor información, puede consultar la página web de Parallax y su sección de aplicaciones para clases: **http://www.stampinclass.com** en el aparte para robótica (*robotics*) y algunos programas comentados que tenemos en nuestra página web: **www.cekit.com.co**

Parallax y Basic Stamp II son marcas registradas de Parallax Inc.

PROYECTO 31



TERMÓMETRO DIGITAL *JUMBO*

La temperatura es una variable física que está involucrada en casi todas las actividades humanas, desde las aplicaciones industriales, hasta la forma en que afecta nuestras actividades diarias. Por esta razón, es frecuente encontrar muchos sitios en los que se han instalado termómetros de gran tamaño que permiten ser vistos a distancia.

Características

- Intervalo de medición entre –10 y +70 grados centígrados
- Visualización en *displays* gigantes construidos con diodos LED
- Utiliza sensor de temperatura en circuito integrado
- Microcontrolador 16C71

El mundo real se rige por fenómenos y leyes físicas cuyas magnitudes toman valores continuos a través del tiempo, esto quiere decir que se comportan de forma analógica. Dichos fenómenos y las variables que los representan, son de mucho interés para nosotros ya que ellas indican el estado en que se encuentra todo aquello que nos rodea. Una de las más importantes es la temperatura, la cual está involucrada en muchos aspectos de la vida humana.

En este proyecto, vamos a desarrollar un termómetro digital, el cual permite medir la temperatura en una gama que va desde los 10º bajo cero hasta los 70º C, una gama más que suficiente para un buen número de tareas. Una de las características importantes de este proyecto es que el valor medido se muestra en *displays* de gran tamaño, lo que lo hace muy atractivo.

Aplicaciones

Este instrumento puede ser utilizado para medir la temperatura ambiente en cualquier oficina, industria, o en el hogar. También puede ser utilizado en laboratorios de química y física ya que la posibilidad de medir temperaturas bajo 0º C le da gran versatilidad.

Descripción

El *termómetro digital jumbo* recibe su nombre debido a que el valor medido se presenta en *displays* de gran tamaño, además, está montado en un chasis diseñado para ser instalado en la pared, de tal forma que pueda ser visto desde muchos ángulos. Posee signo negativo, el cual se enciende cuando la lectura está por debajo de 0º C. Este proyecto se puede conseguir en forma de *kit* bajo la referencia **K-172** de CEKIT.

Funcionamiento

El circuito consta básicamente de cuatro bloques, el primero de ellos, es el sensor de temperatura y la parte análoga de acondicionamiento de señal, el segundo, es el circuito de control basado en un microcontrolador PIC16C71, el tercero corresponde a la fuente de alimentación del sistema y el cuarto, a los elementos de visualización (*displays*). El diagrama esquemático del circuito se muestra en la **figura 31.1**.

Sensor de temperatura

El dispositivo utilizado como sensor es un LM335, **figura 31.2**, el cual tiene una variación en su salida de 10mV/ºK. Esto quiere decir que cuando se tenga una temperatura de 0º C, equivalentes a 273º Kelvin, el sensor tendrá en su salida 2,73 voltios. Este componente posee tres pines (+, - y ADJ), en este caso sólo vamos a utilizar dos de ellos (+ y -).

En la parte sombreada de la **figura 31.1** se muestran la forma de conectar el sensor de temperatura y la parte del circuito que hace el acondicionamiento de señal, la cual se ha implementado alrededor de un amplificador operacional LM358. Este circuito requiere una calibración especial en varios puntos antes de ponerlo en operación, por lo tanto se debe prestar atención a las siguientes instrucciones:

- El primer amplificador está configurado como diferenciador (restador) de ganancia 1, en cuya entrada positiva se ha conectado el LM335 y en la negativa se ha conectado el punto central del potenciómetro R3, este último sirve para fijar un voltaje de referencia de 2,0 voltios, el cual se puede medir entre tierra y el punto de prueba marcado TP1. El objeto de esta parte del circuito es restarle 2 voltios al valor entregado por el LM335; de esta forma, si el LM335 entrega 2,95 voltios (equivalentes a 22º C), en la salida del amplificador operacional (pin 7) se deben medir 0,95 voltios. Para verificarlo se ha dispuesto un punto de prueba marcado TP2 a la salida del operacional.

- El segundo amplificador operacional se configura como amplificador no inversor, cuya ganancia se ajusta con el potenciómetro R11. El factor de amplificación buscado es de 1,5, para lograrlo, se debe medir en el punto TP2 el ni-

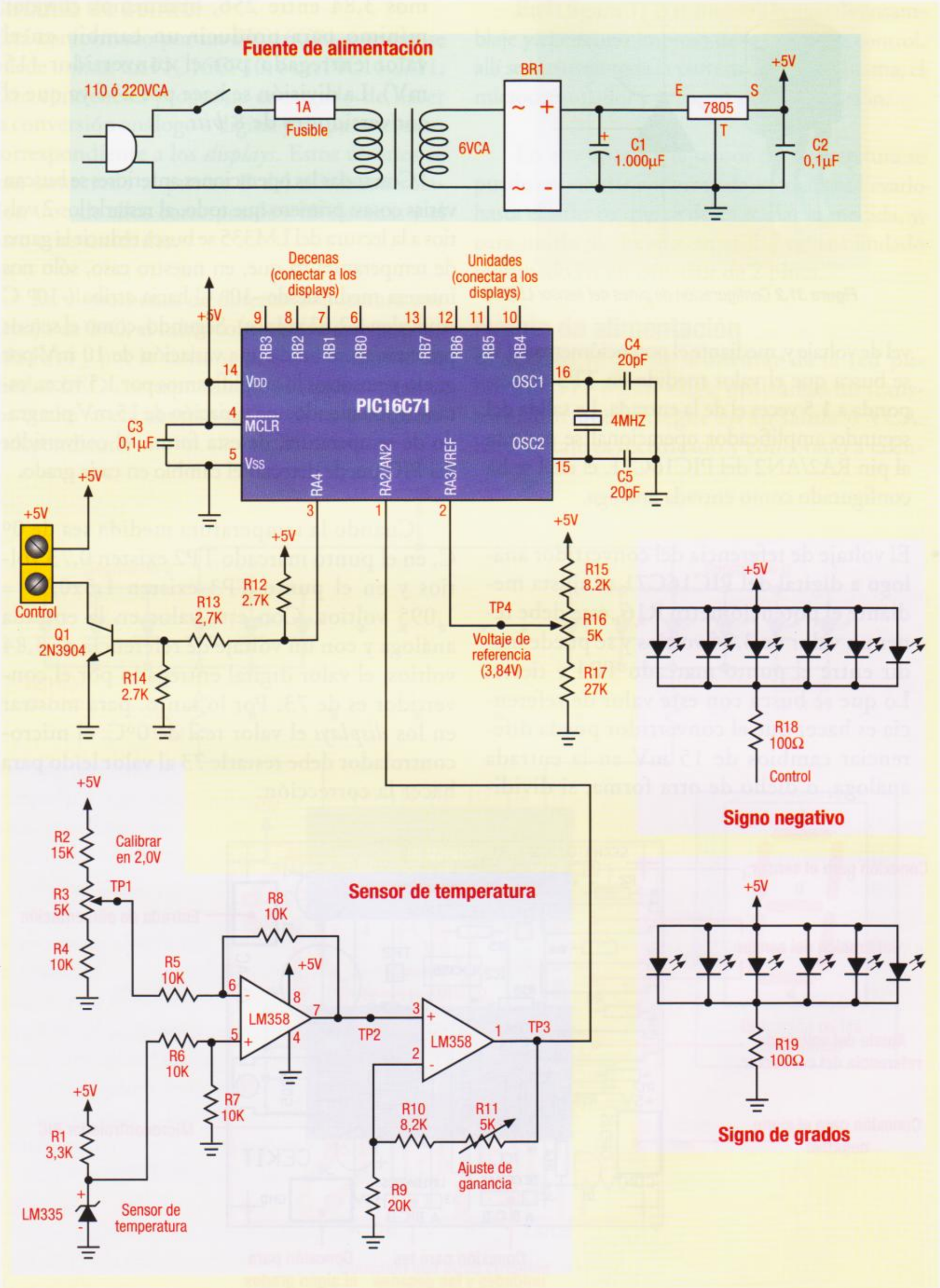


Figura 31.1 Diagrama esquemático del termómetro digital jumbo

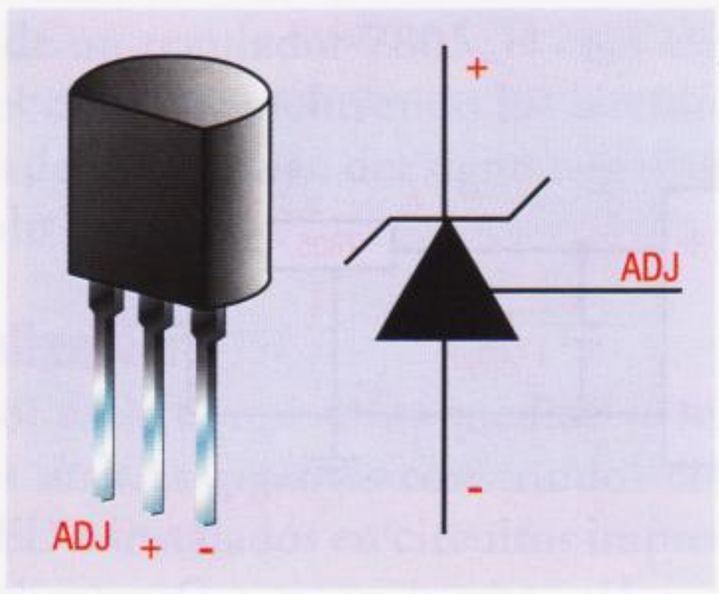


Figura 31.2 *Configuración de pines del sensor LM335*

vel de voltaje y, mediante el potenciómetro R11, se busca que el valor medido en TP3 corresponda a 1,5 veces el de la entrada. La salida del segundo amplificador operacional se conecta al pin RA2/AN2 del PIC16C71, el cual se ha configurado como entrada análoga.

- El voltaje de referencia del convertidor análogo a digital del PIC16C71 se ajusta mediante el potenciómetro R16, éste debe tener un valor de 3,84 voltios y se puede medir entre el punto marcado TP4 y tierra. Lo que se busca con este valor de referencia es hacer que el convertidor pueda diferenciar cambios de 15 mV en la entrada análoga, o dicho de otra forma, si dividimos 3,84 entre 256, obtenemos el valor mínimo para producir un cambio en el valor entregado por el convertidor (15 mV). La división se hace por 256 ya que el convertidor es de 8 *bits*.

Con todas las operaciones anteriores se buscan varias cosas: primero que todo, al restarle los 2 voltios a la lectura del LM335 se busca reducir la gama de temperatura ya que, en nuestro caso, sólo nos interesa medir desde -10º C hacia arriba (-10º C equivalen a 2,63 voltios). Segundo, como el sensor presenta en su salida una variación de 10 mV por grado y nosotros lo amplificamos por 1,5 veces, estamos obteniendo una variación de 15 mV por grado de temperatura, de esta forma, el convertidor del PIC puede detectar el cambio en cada grado.

Cuando la temperatura medida sea de 0º C, en el punto marcado TP2 existen 0,73 voltios y en el punto TP3 existen 1,5x0,73 = 1,095 voltios. Con este valor en la entrada análoga y con un voltaje de referencia de 3,84 voltios, el valor digital entregado por el convertidor es de 73. Por lo tanto, para mostrar en los *displays* el valor real de 0ºC, el microcontrolador debe restarle 73 al valor leído para hacer la corrección.

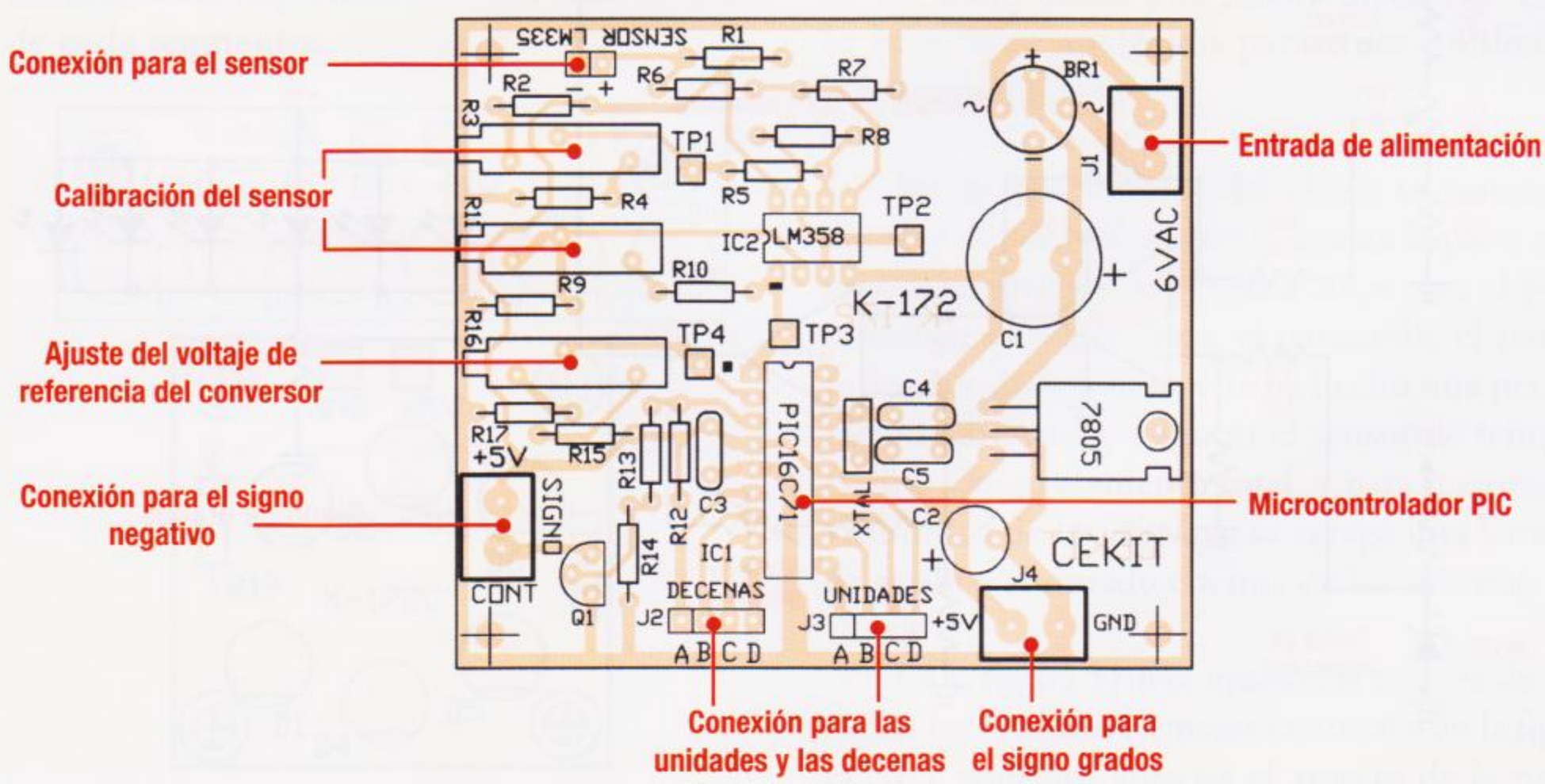


Figura 31.3 *Guía de ensamblaje del termómetro digital*

Circuito de control

Está conformado por un PIC16C71, aunque se puede utilizar un PIC16C710, ó un PIC16C711. Éste contiene el programa encargado de hacer la conversión análogo a digital y de pasar el dato correspondiente a los *displays*. Estos últimos se han conectado al puerto B del microcontrolador, uno de ellos corresponde a las decenas y el otro a las unidades.

En el momento de obtener una lectura por debajo de 0ºC, el microcontrolador activa su salida RA4 y por lo tanto se activa el conjunto de diodos LED correspondiente al signo negativo, los cuales se encuentran en un circuito impreso independiente.

En la **figura 31.3** se muestra la guía de ensamblaje y el circuito impreso de la tarjeta de control, allí se incluyen toda la parte análoga del sistema, el microcontrolador y la fuente de alimentación.

En este aparato, el sensor de temperatura se puede ensamblar en forma de sonda para llevarlo hasta el sitio en que se desea tomar la medida, y, para unirlo al circuito, se utiliza cable blindado terminado en un conector de 2 pines.

Fuente de alimentación

El sistema se puede alimentar de la red pública de 110 ó 220 VCA, utilizando un transformador que entregue en su salida 6 VCA. Este voltaje es rectificado y conectado a la en-

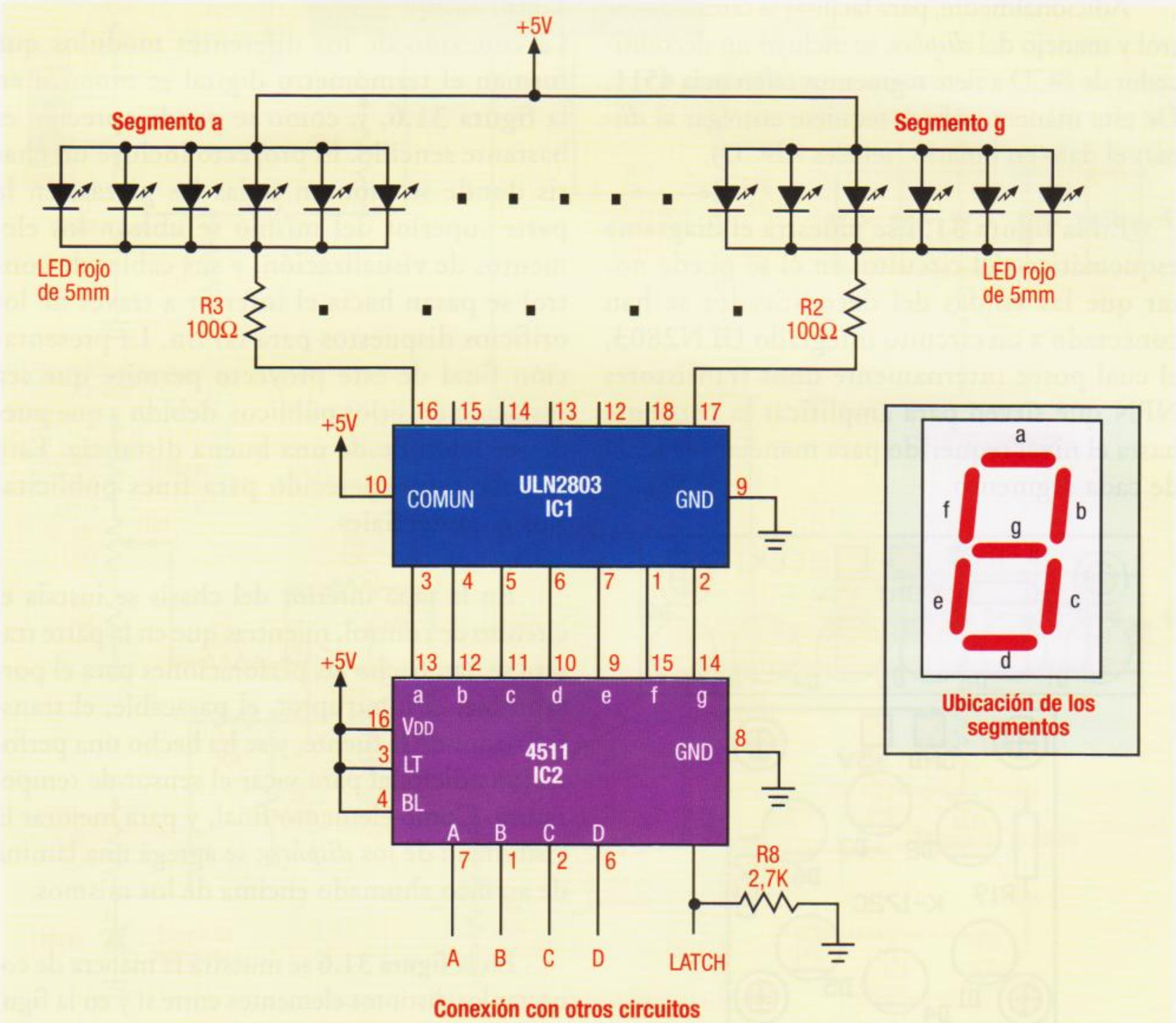


Figura 31.4 *Diagrama esquemático del display de diodos LED*

trada de un regulador 7805, el cual alimenta todo el circuito, incluyendo los circuitos impresos de los *displays*, del signo negativo y del símbolo de grados.

Visualización

El valor de la temperatura medida se muestra en dos *displays* gigantes construidos con diodos LED e instalados en circuitos impresos independientes. Con estos circuitos se busca crear un *display* de siete segmentos pero con la particularidad de ser más grande de lo normal, para ello se diseñó un circuito impreso en el que cada uno de los segmentos está formado por seis LED rojos de 5mm.

Adicionalmente, para facilitar la tarea de control y manejo del *display*, se incluyó un decodificador de BCD a siete segmentos referencia 4511. De esta manera, sólo se requiere entregar al *display* el dato en binario (señales ABCD).

En la **figura 31.4** se muestra el diagrama esquemático del circuito, en él se puede notar que las salidas del decodificador se han conectado a un circuito integrado ULN2803, el cual posee internamente unos transistores NPN que sirven para amplificar la corriente hasta el nivel requerido para manejar los LED de cada segmento.

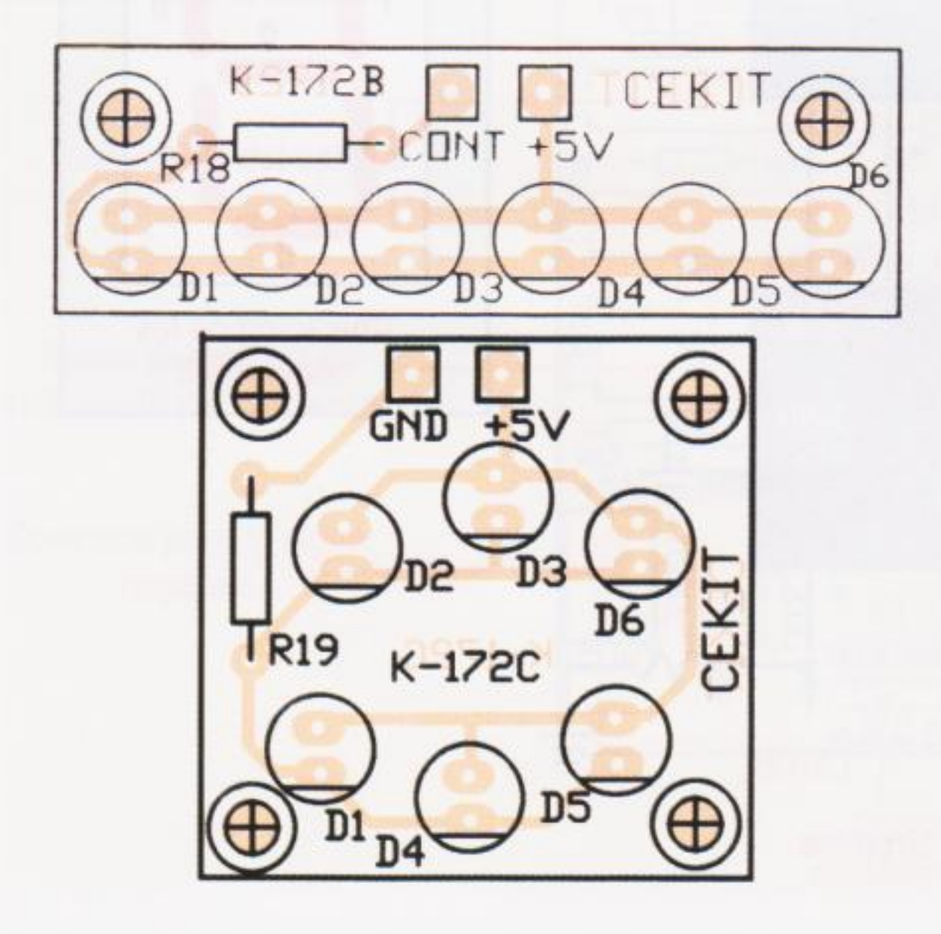


Figura 31.5 *Circuito impreso de los signos negativo y de grados*

El microcontrolador entrega a cada circuito de visualización los *bits* que se conectan en las entradas ABCD de los mismos. Estos circuitos requieren, además, de una alimentación de +5V.

El signo negativo se debe conectar al circuito de control en los puntos marcados con la palabra *signo*, allí se encuentra un borne con la fuente de +5V, y otro, con la señal de control que indica el momento en que se debe encender. El signo de grados (°), se conecta a la fuente de alimentación de 5V directamente. En la **figura 31.5** se muestran los circuitos impresos de estos módulos.

Ensamblaje

La conexión de los diferentes módulos que forman el termómetro digital se muestra en la **figura 31.6**, y, como se puede apreciar, es bastante sencillo. El proyecto incluye un chasis donde se montan todas las piezas, en la parte superior del mismo se ubican los elementos de visualización, y sus cables de control se pasan hacia el interior a través de los orificios dispuestos para tal fin. La presentación final de este proyecto permite que sea instalado en sitios públicos debido a que puede ser leído desde una buena distancia. Esto lo hace muy apetecido para fines publicitarios o comerciales.

En la tapa inferior del chasis se instala el circuito de control, mientras que en la parte trasera se han hecho las perforaciones para el portafusible, el interruptor, el pasacable, el transformador de la fuente, y se ha hecho una perforación adicional para sacar el sensor de temperatura. Como elemento final, y para mejorar la visibilidad de los *displays*, se agrega una lámina de acrílico ahumado encima de los mismos.

En la **figura 31.6** se muestra la manera de conectar los distintos elementos entre sí y en la **figura 31.7** podemos apreciar el aspecto de la parte trasera del chasis con los elementos ya instalados.

DECENAS

UNIDADES

SÍMBOLO DE GRADOS

SENSOR LM335

6VCA/1A

Fusible 1A

Interruptor

Transformador

110 ó 220VCA

SIGNO NEGATIVO

Figura 31.6 Conexión de los diferentes elementos

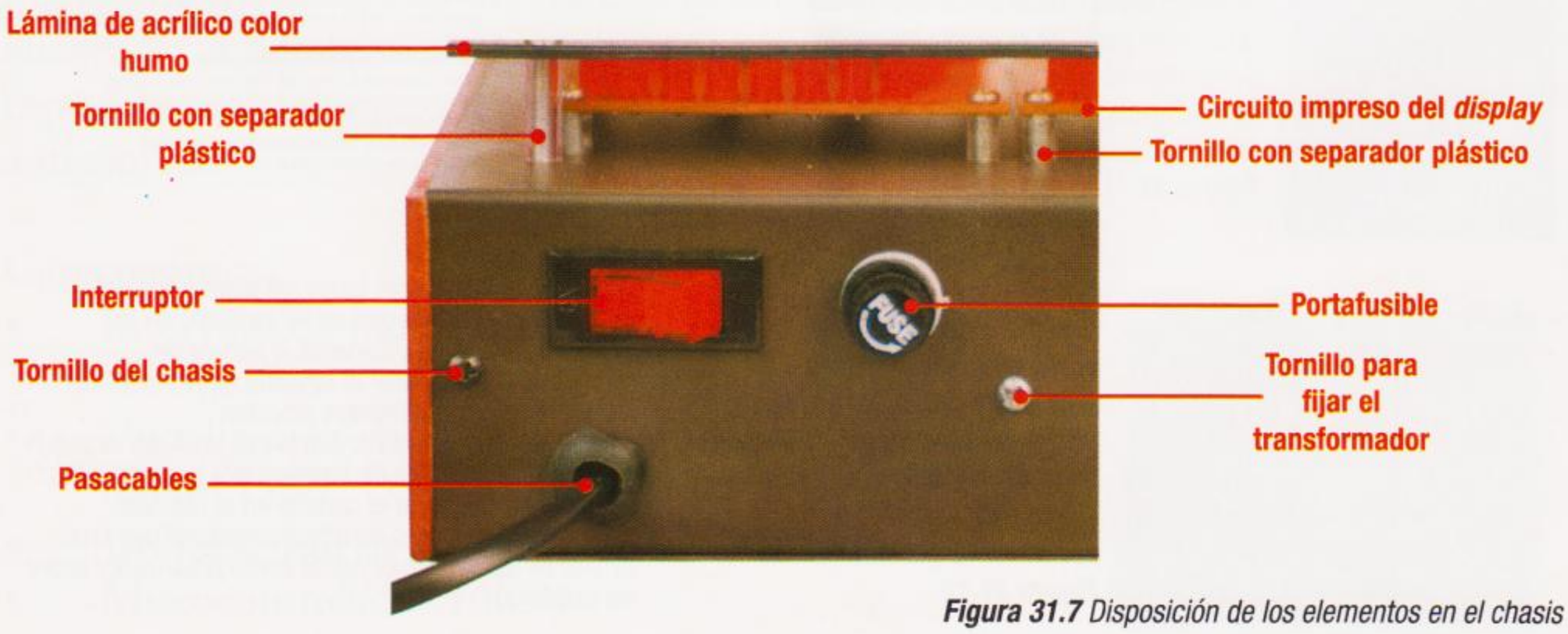


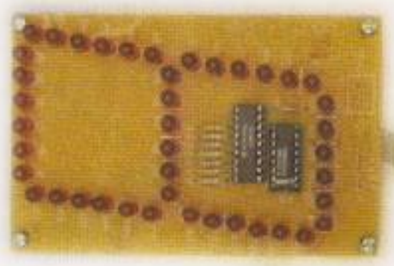
Figura 31.7 Disposición de los elementos en el chasis

Ensamblaje del proyecto

Figura 31.8 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el termómetro digital jumbo, CEKIT K-172.*

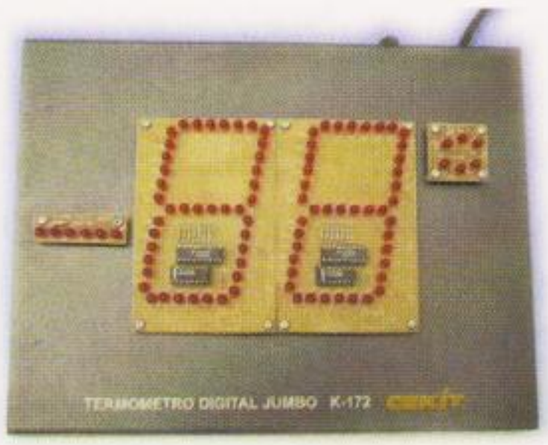
Con este circuito se construye un termómetro que mide con precisión de 1°C, la temperatura en grados centígrados entre -55° y 99°C, y usa dos displays grandes (jumbo) para su visualización. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

Primero se instalan y se sueldan todos los diodos LED y las resistencias necesarias para ensamblar las tarjetas de los signos menos y de grados.
Figura 31.9



Luego se instalan y sueldan las resistencias, los diodos LED, las bases para los circuitos integrados y el conector, que conforman la tarjeta del display jumbo.
Figura 31.10

Instale y suelde todos los componentes que hacen parte de la tarjeta de control del termómetro jumbo, y asegúrela con tornillos y separadores plásticos sobre la tapa inferior del chasis metálico.
Figura 31.11



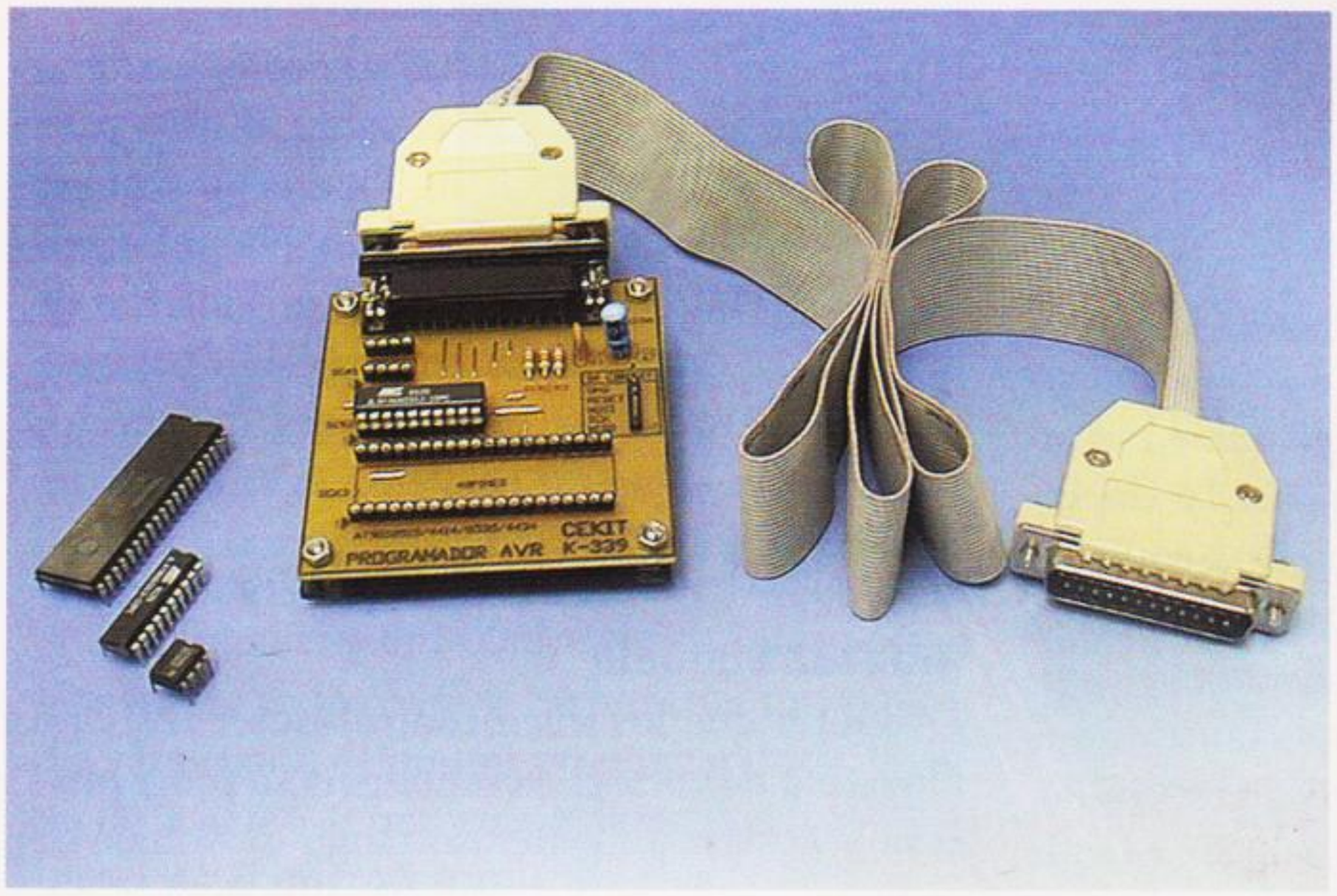
Luego de ensamblar las tarjetas, asegúrelas con tornillos y separadores plásticos en la cara frontal del chasis; y efectúe las conexiones necesarias dentro del chasis y de acuerdo con los esquemas de conexión.
Figura 31.12

LISTA DE MATERIALES

Cant.	Ref.	Descripción
Display jumbo con LED K-170A (lista para uno)		
1	Circuito integrado ULN2803	
1	IC2	Circuito integrado CD4511
7	R1 a R7	Resistencias de 100 Ohm a 1/4 W
1	R8	Resistencia de 2,7K a 1/4 W
42	D1 a D42	Diodos LED rojos de 5 mm
1	K-170A	Circuito impreso CEKIT
1		Conector en línea de 4 pines (completo)
1		Base para integrado de 18 pines
1		Base para integrado de 16 pines
30cm		Cable polarizado 2x20 ó 2x22AWG
30cm		Cable ribbon de 4 hilos
30cm		Alambre telefónico para puentes
4		Tornillos milimétricos de 3x15 con sus tuercas
4		Separadores plásticos de 1cm
Circuito de control del termómetro K-172		
1	IC1	Microcontrolador PIC16C71, grabado
1	IC2	Circuito integrado LM358
1	IC3	LM335 (Sensor de temperatura)
1	REG	Regulador fijo de +5V LM7805
1	Q1	Transistor 2N3904
1	BR1	Puente rectificador circular de 1,5 A (W04M)
1	R1	Resistencia de 3,3K a 1/4 W
1	R2	Resistencia de 15K a 1/4 W
5	R4 a R8	Resistencias de 10K a 1/4 W
1	R9	Resistencia de 20K a 1/4 W
2	R10 y R15	Resistencias de 8,2K a 1/4 W
3	R12 a R14	Resistencias de 2,7K a 1/4 W
1	R17	Resistencia de 27K a 1/4W
3	R3, R11 y R16	Trimmers resistivos multivueltas de 5 K
1	C1	Condensador electrolítico de 1.000µF/25V
1	C2	Condensador electrolítico de 10µF/25V
1	C3	Condensador cerámico de 0,1 µF
2	C4 y C5	Condensadores cerámicos de 20 pF
1	XTAL	Cristal de cuarzo de 4MHz
1	K-172	Circuito impreso CEKIT
3		Conectores de tornillo de 2 pines
1		Conector en línea de 2 pines (completo)
1		Base para integrado de 18 pines
1		Base para integrado de 8 pines
1		Disipador tipo TO-220 (el del K-025)
1		Tornillo milimétrico de 3x7 con su tuerca
4		Tornillos milimétricos de 3x15 con sus tuercas
4		Separadores plásticos de 1cm
50cm		Cable blindado monofónico para el sensor
3cm		Termoencogible de 3mm
Signo negativo K-172B		
6	D1 a D6	Diodos LED rojos de 5 mm
1	R18	Resistencia de 100 Ohm a 1/4 W
1	K-172B	Circuito impreso CEKIT
2		Tornillos milimétricos de 3x15 con sus tuercas
2		Separadores plásticos de 1cm
30cm		Cable polarizado 2x20 ó 2x22AWG
Signo de grados K-172C		
6	D1 a D6	Diodos LED rojos de 5 mm
1	R19	Resistencia de 100 Ohm a 1/4 W
1	K-172C	Circuito impreso CEKIT
4		Tornillos milimétricos de 3x15 con sus tuercas
4		Separadores plásticos de 1cm
30cm		Cable polarizado 2x20 ó 2x22AWG
Varios		
1	K-172	Chasis metálico
1		Transformador M-508 (Prim:110 ó 120VCA, Sec:6VCA, Cap:1A)
1m		Cable polarizado 2x20 ó 2x22AWG
1		Cable de potencia con enchufe
1		Portafusible corto para chasís
1		Fusible corto de 1A
1		Pasacable de caucho
1		Interruptor de balancín con luz piloto
1		Lámina de acrílico ahumado (27 x 15 cm y perforada)
4		Tornillos milimétricos de 3x25 con sus tuercas (negros)
2		Tornillos milimétricos de 3x7 con sus tuercas (negros)
10		Tornillos golosos para lámina (negros)
1		Cable ribbon de 4 hilos (60 cm) (para los displays)
1		Jack monofónico
1		Plug monofónico
4		Terminales para circuito impreso (espadines)
4		Separadores plásticos de 2 cm

Pruebas finales

Después de ensamblar todos los elementos del proyecto en el orden que se ha indicado en las imágenes anteriores, conecte el sensor de temperatura y alimente el circuito; debe visualizar la temperatura en los displays grandes.
A partir de este momento basta con someter la sonda del sensor a cambios de temperatura para visualizar de manera inmediata el cambio en la medida.
Para obtener mejores resultados visuales use una lámina de acrílico color humo como polarizador sobre los displays.



PROGRAMADOR PARA MICROCONTROLADORES ATMEL

En esta ocasión construiremos un sencillo programador para microcontroladores ATMEL AT90Sxxxx y analizaremos al manejo de algunas herramientas de *software* apropiadas para el rápido desarrollo de programas con esta familia de microcontroladores.

Características

- No requiere fuente de alimentación
- Bajo costo
- Fácil de construir
- Incluye programa de interfaz con la computadora
- Trabaja en cualquier computadora tipo PC, desde XT hasta Pentium
- Soporta microcontroladores de la serie AVR

Microcontroladores que acepta
AT90S2323 (8 pines)
AT90S2343 (8 pines)
AT90S2313 (20 pines)
AT90S8515 (40 pines)
AT90S4414 (40 pines)
AT90S8535 (40 pines)
AT90S4434 (40 pines)

Importancia de los dispositivos programadores

A lo largo de este curso hemos estudiado algunos dispositivos programadores para las distintas familias de microcontroladores y hemos resaltado su importancia para sentar bases reales y concretas de nuestro aprendizaje en este campo de la electrónica. Lo más importante es que hemos construido sistemas muy económicos, fáciles de elaborar ,y, lo más importante, con toda la información necesaria para su construcción, además, todo el *software* disponible para su funcionamiento puede ser descargado de nuestra página **www.cekit.com.co**.

Un elemento de este tipo constituye una de las principales herramientas que debemos tener en nuestro laboratorio, pues si no se cuenta con ella, no podremos desarrollar sistemas microcontrolados.

Ahora analicemos un sencillo programador para los microcontroladores ATMEL. Dispositivos como éste se encuentran disponibles en la red para ser descargados y utilizados sin ningún problema; el que mostramos a continuación se escogió debido a su buen diseño y fácil construcción.

Este dispositivo y su *software* fueron elaborados por el ingeniero Jerry Meng de Beijing (China), quien lo dispuso en modo de *freeware* para ser copiado libremente.

Programador por el puerto paralelo

El puerto paralelo de las computadoras es un dispositivo muy versátil. A pesar de su nombre, es posible utilizarlo para manejar datos serialmente aún a mayor velocidad que un puerto serial convencional.

En los programadores anteriores hemos utilizado tanto el puerto serial como el puerto paralelo para conectar las tarjetas a la computadora, cada método tiene ventajas sobre el otro pero ambos son igual de efectivos. En este proyecto se utiliza el puerto paralelo para programar los microcontroladores AVR utilizando una interfaz de muy pocos cables y una construcción bastante sencilla. Lo único que se necesita es un conector DB-25 para el puerto paralelo, un *socket* o base de 20 ó 40 pines y el circuito impreso.

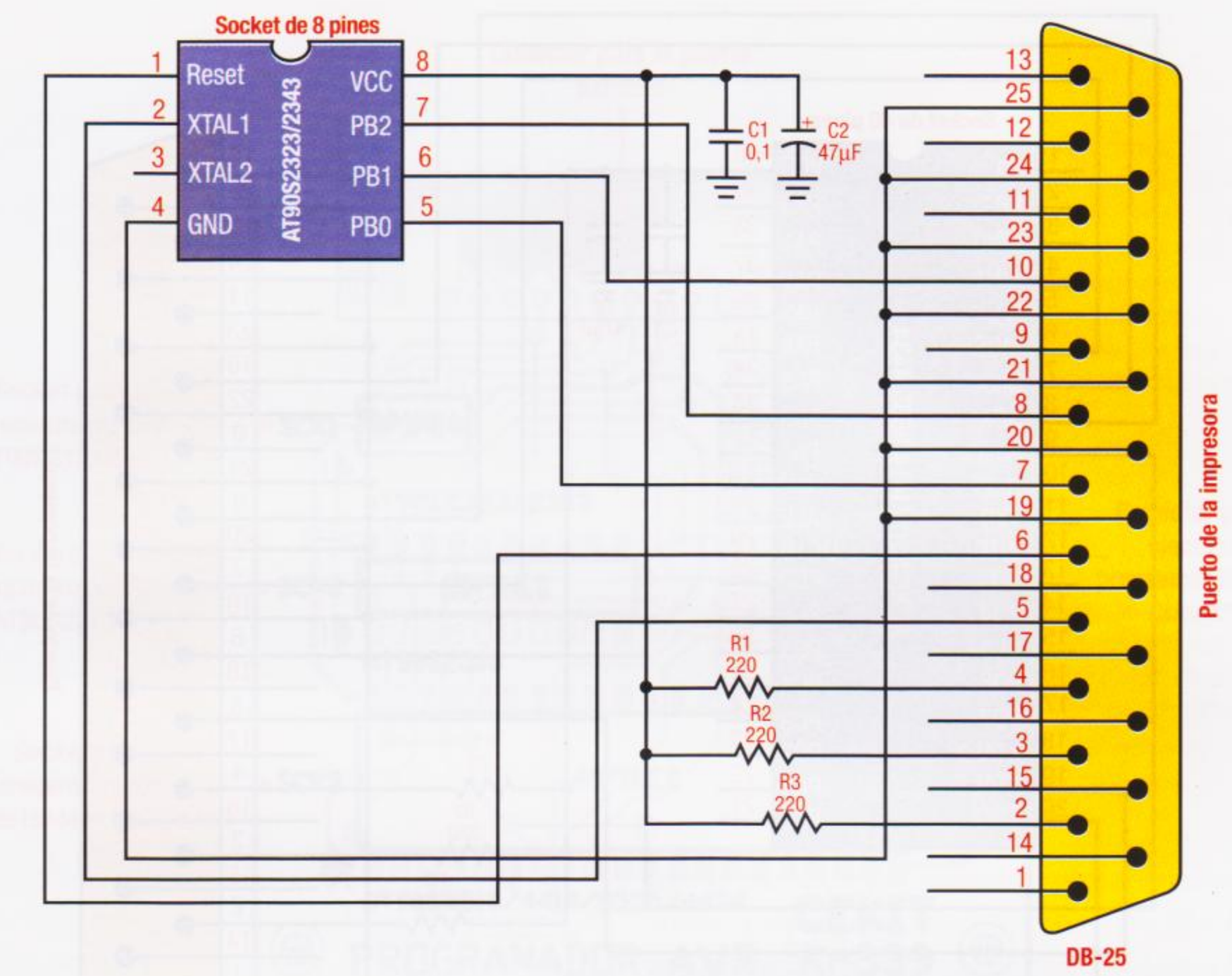
Interfaz sencilla

Debido a que los microcontroladores AVR no requieren de voltajes elevados para su programación y poseen un generoso rango de voltaje de trabajo (2,7V – 6V), es posible alimentar el circuito con los pines de datos del puerto paralelo.

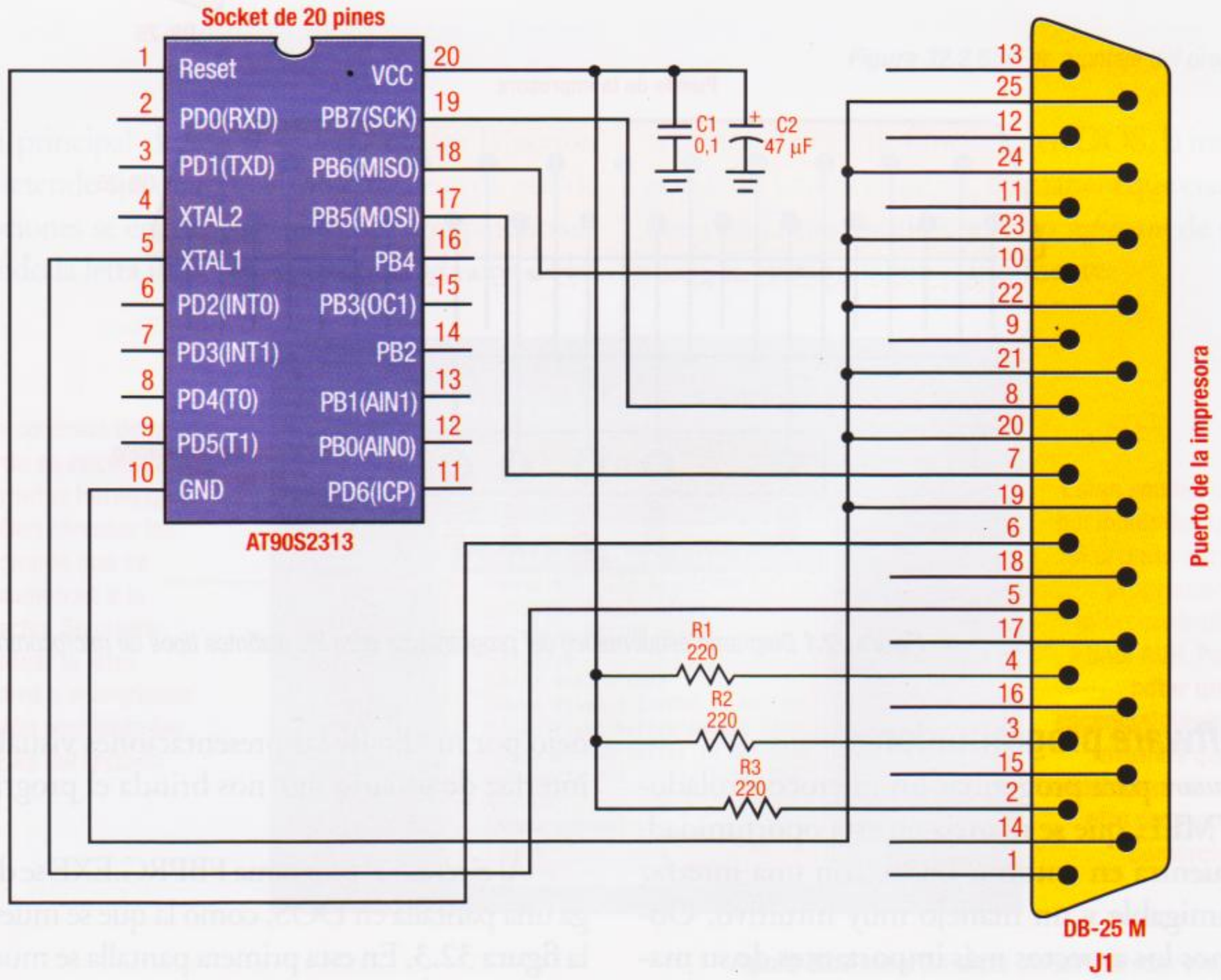
Recordemos que el puerto paralelo (de 25 pines) que se encuentra en cada computadora, posee ocho *bits* para la salida de datos, cinco *bits* de estado y cuatro *bits* de salidas de control. Las computadoras más modernas poseen *bits* bidireccionales, pero esa característica no es utilizada en este caso.

En la **figura 32.1** podemos apreciar el diagrama esquemático de las posibles configuraciones para los distintos tipos de encapsulados de los microcontroladores; como se puede ver, los únicos elementos adicionales son las resistencias de 220 Ohmios. Observemos en la **figura 32.1.d** que el sistema también puede llevar a cabo programación "*in circuit*" conectando la tarjeta al circuito por medio de los pines CLK, MOSI, MISO y GND. En la **figura 32.2** se muestra la guía de montaje del programador.

(a)


Socket de 8 pines
Reset
XTAL1
XTAL2
GND
AT90S2323/2343
VCC
PB2
PB1
PB0
C1
0,1
C2
47µF
R1
220
R2
220
R3
220
Puerto de la impresora
DB-25

(b)


Socket de 20 pines
Reset
PD0(RXD)
PD1(TXD)
XTAL2
XTAL1
PD2(INT0)
PD3(INT1)
PD4(T0)
PD5(T1)
GND
VCC
PB7(SCK)
PB6(MISO)
PB5(MOSI)
PB4
PB3(OC1)
PB2
PB1(AIN1)
PB0(AIN0)
PD6(ICP)
AT90S2313
C1
0,1
C2
47 µF
R1
220
R2
220
R3
220
Puerto de la impresora
DB-25 M
J1

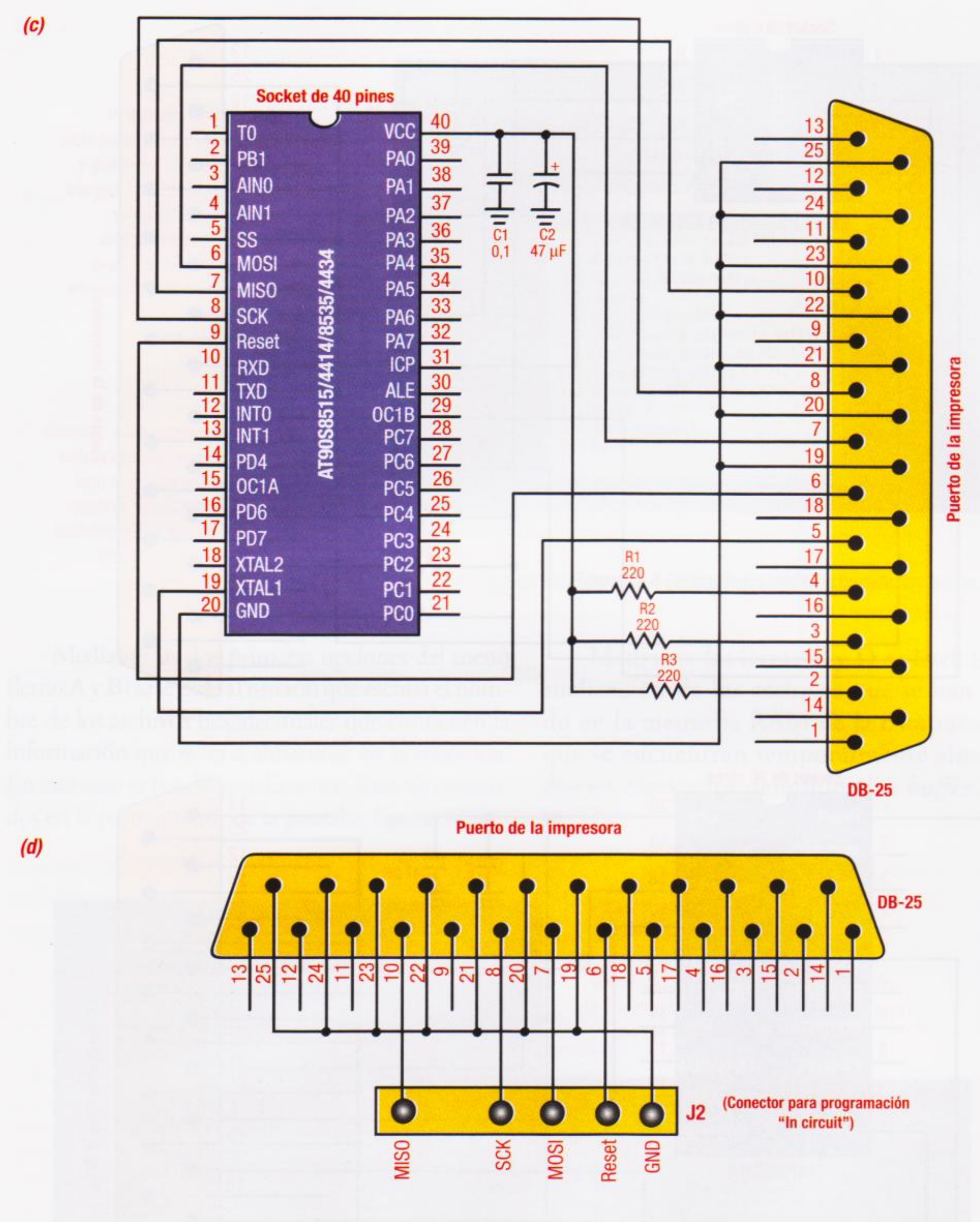


Figura 32.1 *Diagrama esquemático del programador para los distintos tipos de microcontroladores*

El *software* programador

El *software* para programar los microcontroladores ATMEL, que se maneja en esta oportunidad, se encuentra en entorno DOS, con una interfaz muy amigable y un manejo muy intuitivo. Observemos los aspectos más importantes de su manejo por medio de las presentaciones visuales y la interfaz de usuario que nos brinda el programa.

Al ejecutar el programa FBPRG.EXE se despliega una pantalla en DOS, como la que se muestra en la **figura 32.3**. En esta primera pantalla se muestra el

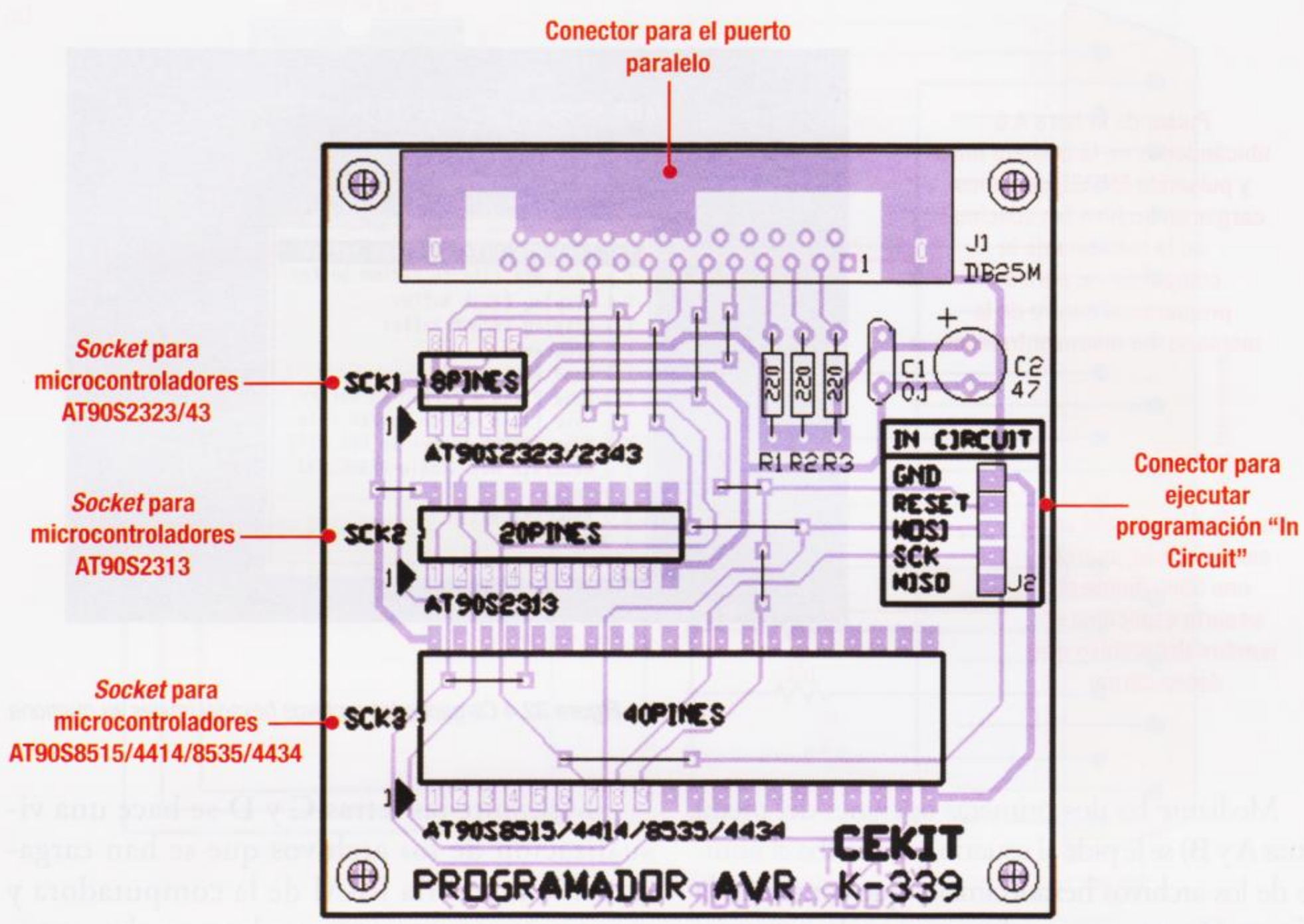


Figura 32.2 *Guía de montaje del programador*

menú principal donde el usuario escoge la acción que pretende ejecutar. A la izquierda de cada una de las opciones se encuentra una letra entre paréntesis; pulsando la letra indicada, se ejecuta cada una de las acciones. A pesar de funcionar en DOS, la interfaz es agradable y muy intuitiva, de manera que cualquiera que esté familiarizado con otro *software* de programación puede adaptarse fácilmente.

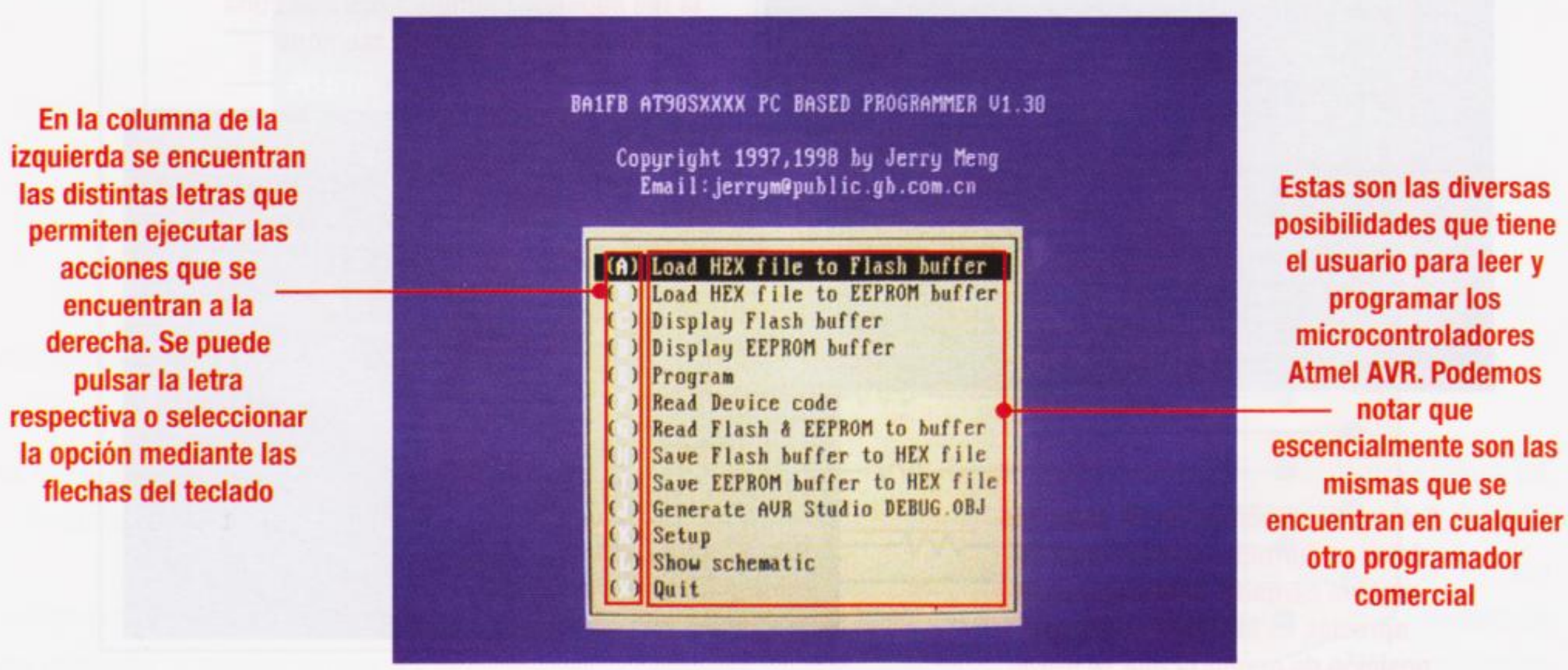


Figura 32.3 *Menú principal del software del programador*

Pulsando la letra A ó ubicándonos en la primera línea y pulsando ENTER, podemos cargar un archivo hexadecimal en la memoria de la computadora, para ser programado dentro de la memoria del microcontrolador

Cuando esta opción se ha seleccionado, aparece una línea donde el usuario especifica el nombre del archivo que desea cargar

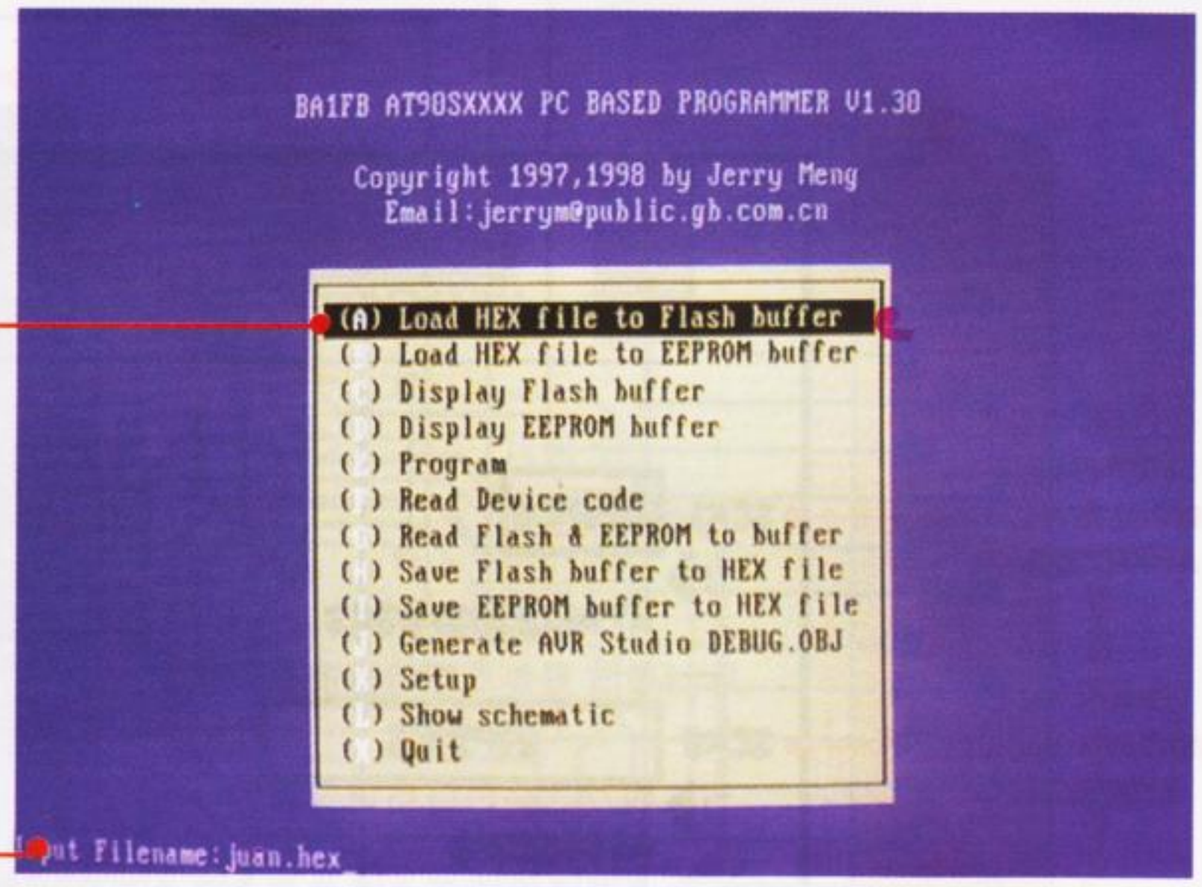
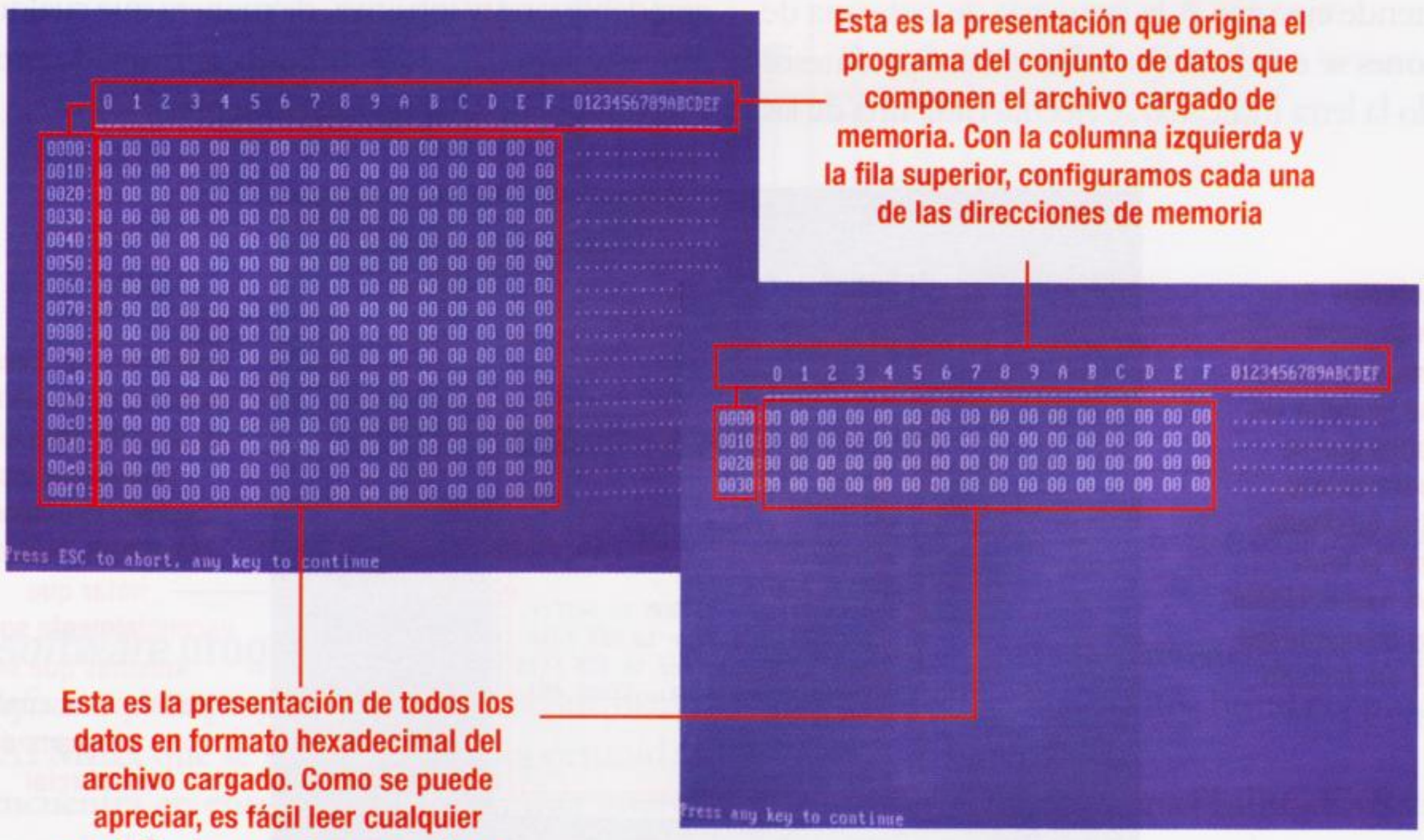


Figura 32.4 *Cargando los archivos hexadecimales en memoria*

Mediante las dos primeras opciones del menú (letras **A** y **B**) se le pide al usuario que escriba el nombre de los archivos hexadecimales que contienen la información que se va a almacenar en la memoria. En este caso se puede visualizar una línea de comandos en la parte inferior de la pantalla, **figura 32.4**.

Mediante las letras **C** y **D** se hace una visualización de los archivos que se han cargado en la memoria RAM de la computadora y que se encuentran temporalmente almacenados en elementos denominados *buffers*, **figura 32.5**.

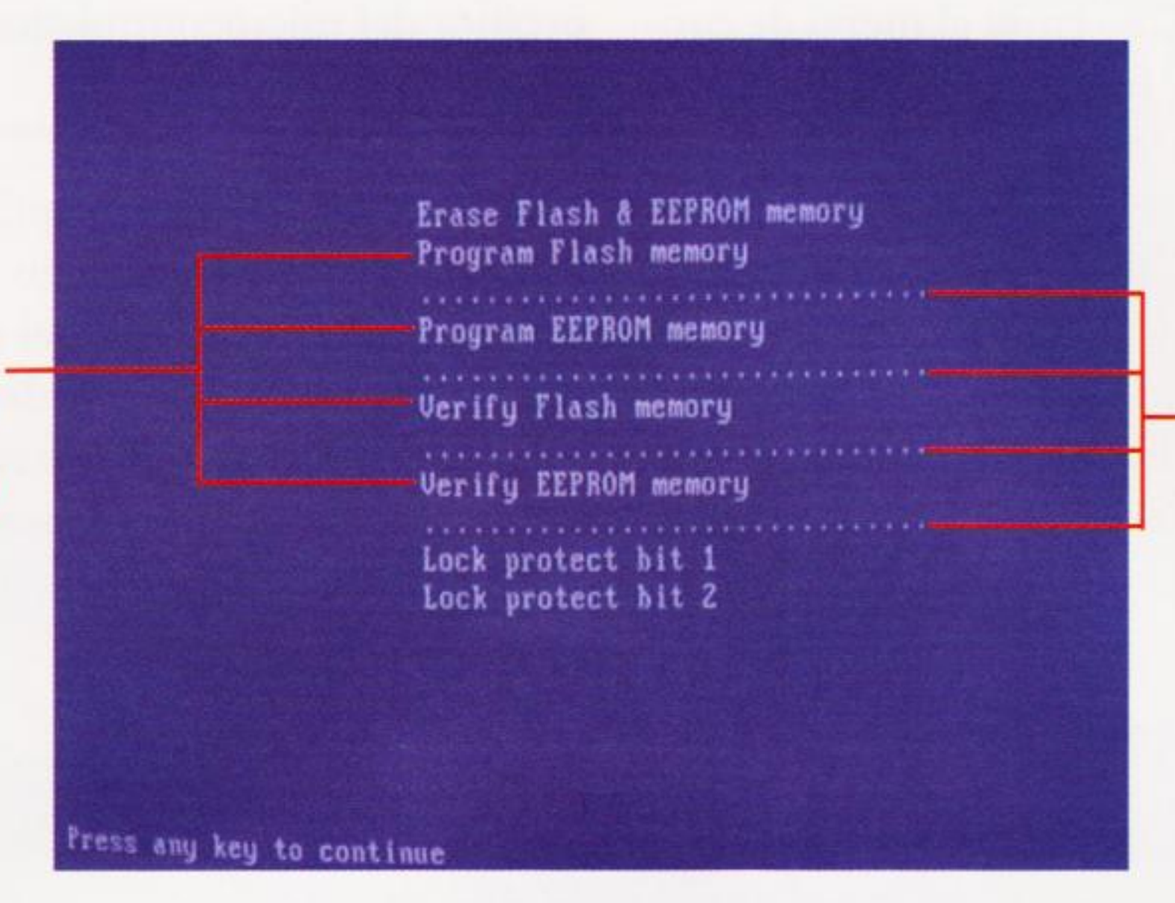
Esta es la presentación que origina el programa del conjunto de datos que componen el archivo cargado de memoria. Con la columna izquierda y la fila superior, configuramos cada una de las direcciones de memoria



Esta es la presentación de todos los datos en formato hexadecimal del archivo cargado. Como se puede apreciar, es fácil leer cualquier posición de memoria que se desee

Figura 32.5 *Visualización de los archivos cargados en memoria*

Cuando ejecutamos desde el Menú principal la opción de programar el microcontrolador aparece esta pantalla, donde podemos apreciar cómo se lleva a cabo dicho proceso en las memorias EEPROM y FLASH



El programa proporciona una forma de indicación del porcentaje de memoria que se ha trabajado (borrado, programado y verificado) que emula una barra progresiva, por medio de puntos. Cada uno de los procesos involucrados en la programación van apareciendo secuencialmente en la pantalla con su respectiva barra.

Figura 32.6 *Programación del microcontrolador*

Cuando se pulsa la letra **E** se procede a programar el microcontrolador con la información contenida en los *buffers*. Se presenta una pantalla y una barra de puntos progresiva que aumenta de acuerdo al porcentaje de memoria programada en el microcontrolador, **figura 32.6**.

El *software* permite verificar los datos programados, esa verificación por lo general se hace comparando los datos de todo el *buffer*, ya sea en el mismo momento de programarse o posteriormente, este procedimiento fue seguido en el programador K-224 y es muy útil para verificar que el microcontrolador se encuentre en buen estado.

Cuando el microcontrolador posea alguna información almacenada, es posible leerla mediante las teclas **F** y **G**, **figura 32.7**. La información contenida en el microcontrolador quedará almacenada en un *buffer* para que pueda ser grabada dentro de un archivo en el disco duro o en un disco flexible, mediante las teclas **H** e **I**, **figura 32.8**.

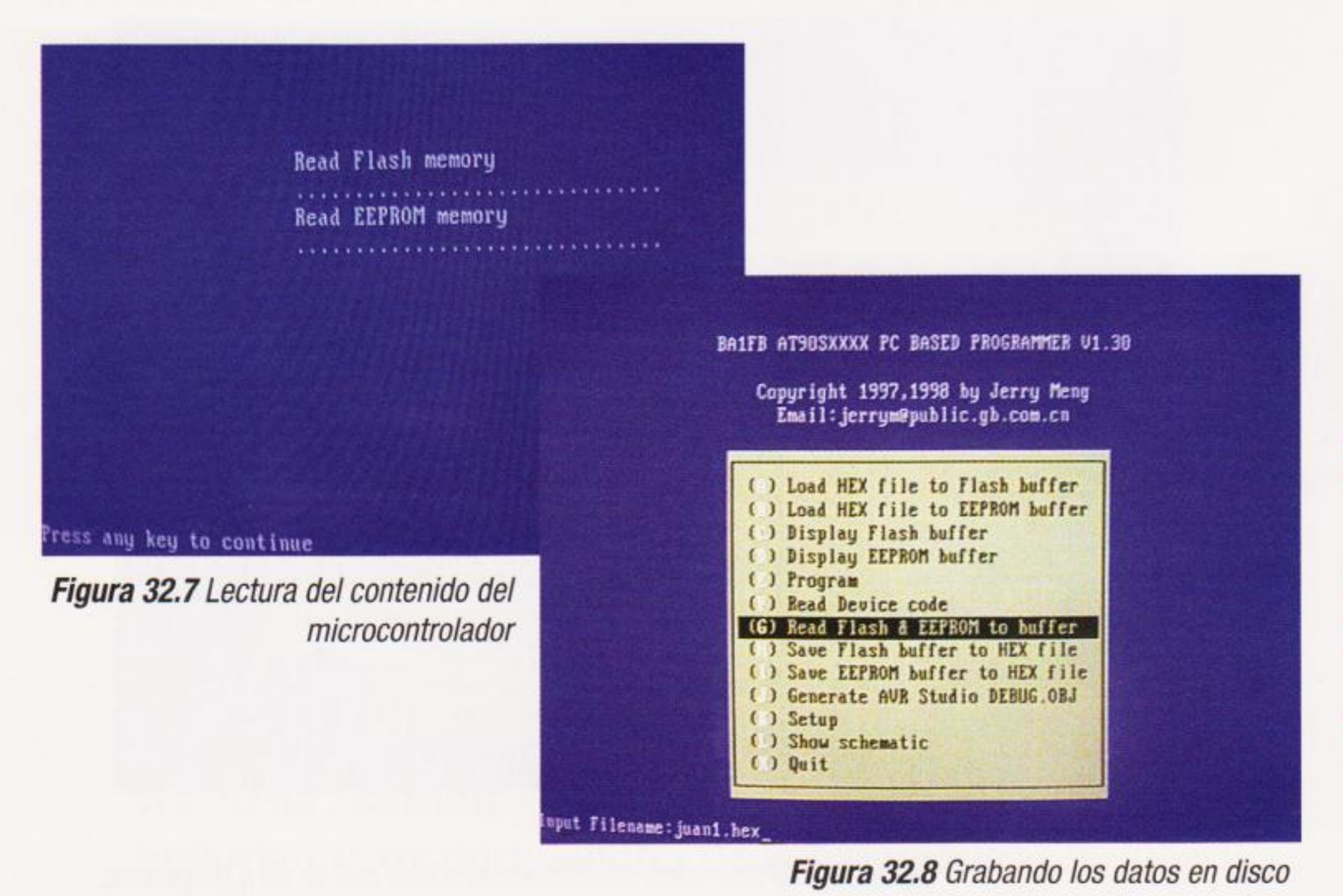


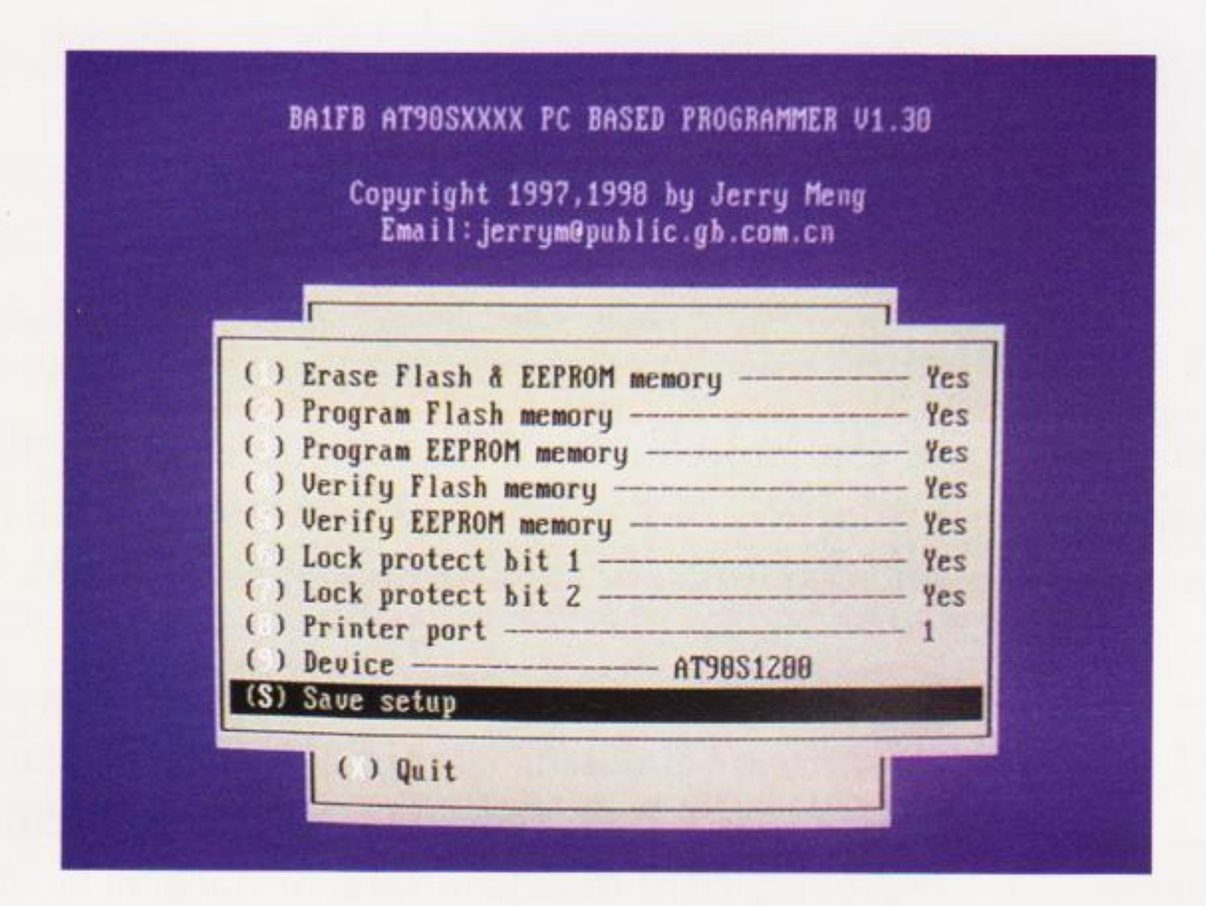
Figura 32.7 *Lectura del contenido del microcontrolador*

Cuando se tiene un microcontrolador programado es posible leer su contenido pulsando las letras F y G del menú principal. La lectura de la memoria también posee su barra indicadora y cuando ella termina podemos grabar esa información en un archivo del disco duro

Figura 32.8 *Grabando los datos en disco*

Pulsando la letra **K** se entra al menú de configuración del sistema, **figura 32.9**. Podemos ver una serie de números entre paréntesis ,y, al frente de cada uno de ellos, cada una de las opciones que el usuario debe establecer. Con cada uno de estos números determinamos las acciones que queremos que se ejecuten sobre el microcontrolador objetivo. Estando en esta pantalla y pulsando el número **9** se selecciona la referencia específica del microcontrolador que se esté trabajando, **figura 32.10**.

El programa nos presenta una pequeña ayuda muy simpática; pulsando la letra **L** se accede al diagrama esquemático del circuito del programador, **figura 32.11**. Puede pensarse que esta ayuda adicional nos permite construir nuevamente la tarjeta en cualquier momento o lugar.



Este es el menú para configurar el funcionamiento del sistema. A la izquierda se encuentra una serie de letras que habilitan o deshabilitan las opciones que se encuentran a la derecha. La última opción sirve para guardar la configuración que ha hecho el usuario

Figura 32.9 *Menú de configuración del sistema*

Pulsando el número 9 se despliega un nuevo menú donde podemos seleccionar el tipo de microcontrolador AVR que vamos a programar. Este nuevo menú posee el mismo formato y método de selección de los anteriores

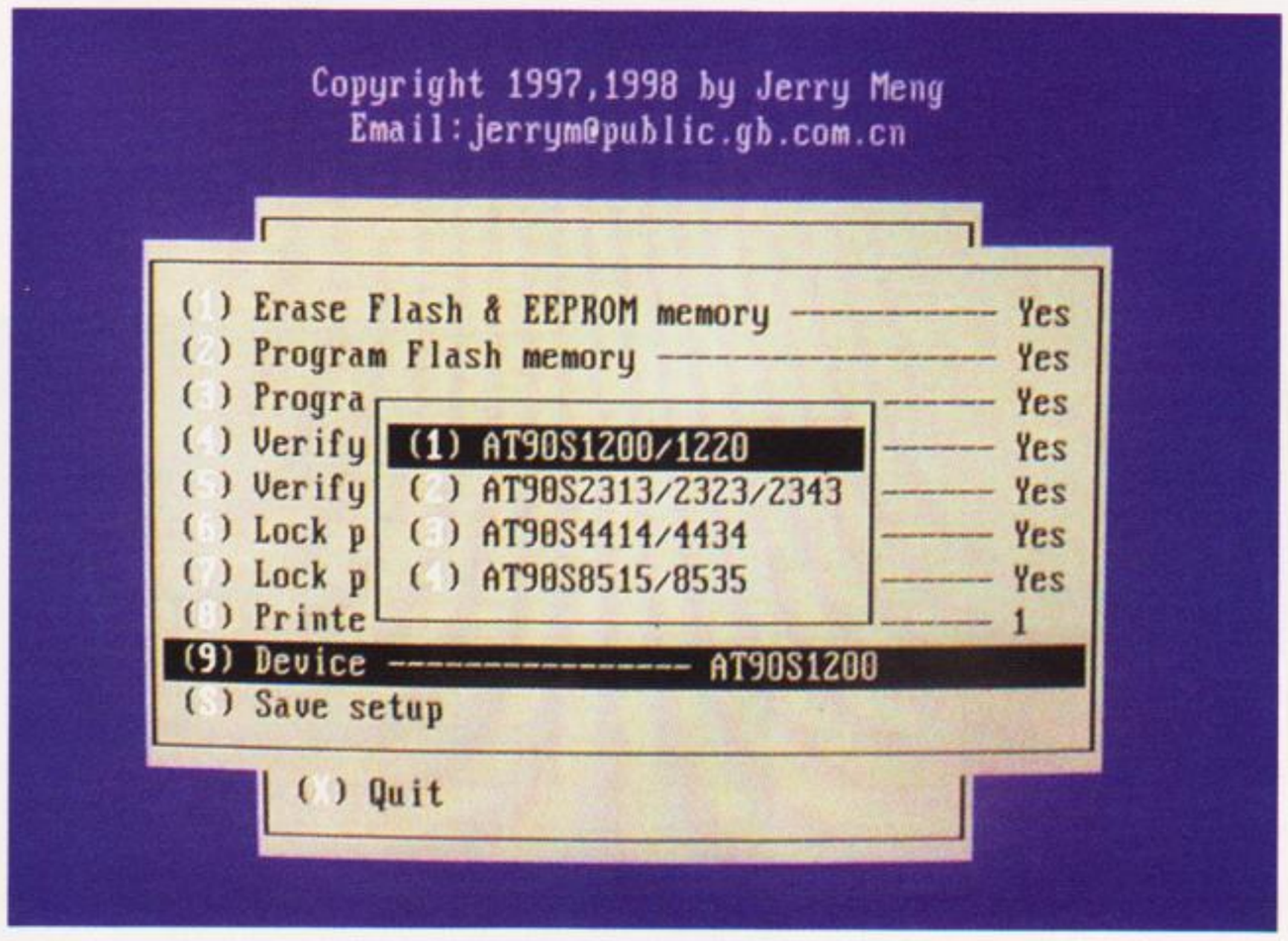


Figura 32.10 *Selección del microcontrolador que se va a programar*

Este es el diagrama esquemático que muestra el programa. Gracias a él podemos montar nuestro programador en cualquier laboratorio o sitio de trabajo. A pesar de que el diagrama se graficó mediante programación en modo texto, tenemos completa claridad de cómo construirlo

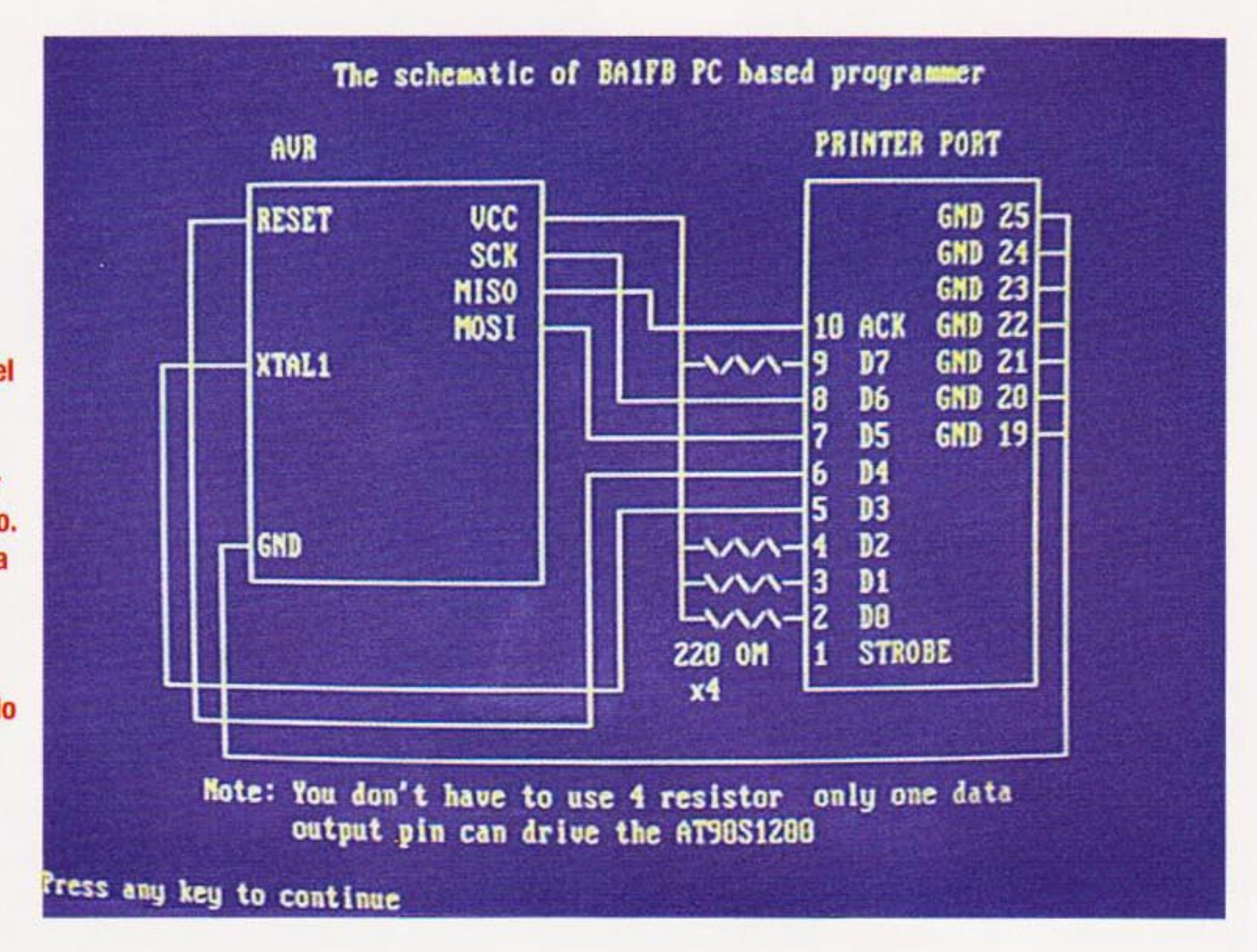


Figura 32.11 *Diagrama esquemático del circuito programador, presentado en el software*

Herramientas de desarrollo adicionales

Los microcontroladores ATMEL están aumentando su nivel de aceptación por sus poderosas características (remitirse a la sección de teoría) y su costo reducido. En la internet ya se han conformado una buena cantidad de páginas, foros y empresas alrededor de estos microcontroladores. Este hecho, sumado a su facilidad de consecución en el mercado, su bajo costo, y la fortaleza mundial en electrónica que está tomando ATMEL, hacen muy llamativa la idea de decidirnos por esta marca.

Visite la página **www.atmel.com** y entérese de todos los productos, soluciones y adelantos a que han llegado los ingenieros de esta prestigiosa empresa.

Muchas empresas de desarrollo electrónico han visto en ATMEL una muy buena oportunidad y han estructurado una serie de herramientas de *hardware* y *software* para distribuir entre los interesados en esta tecnología. Analicemos brevemente dos herramientas de *software* que se pueden descargar libremente de la internet y que constituyen un excelente material de apoyo para los desarrolladores de tecnología con microcontroladores AVR. Ya conocemos la importancia de un buen sistema simulador y las ventajas que brinda el depurar los códigos con un programa especializado de esta naturaleza.

La primera herramienta la proporciona la empresa MICRODESIGN, quien ha desarrollado el programa FASTAVR que es un entorno para la edición y la compilación de programas para microcontroladores de la serie AVR.

La pantalla inicial de este programa se puede apreciar en la **figura 32.12**. Como se puede ver, tiene un editor de texto en el que se escriben los programas donde se numeran cada una las líneas, una barra de herramientas, y una ventana de estado. Cuando se ha escrito el programa podemos compilarlo y crear el archivo hexadecimal que se grabará dentro del microcontrolador.

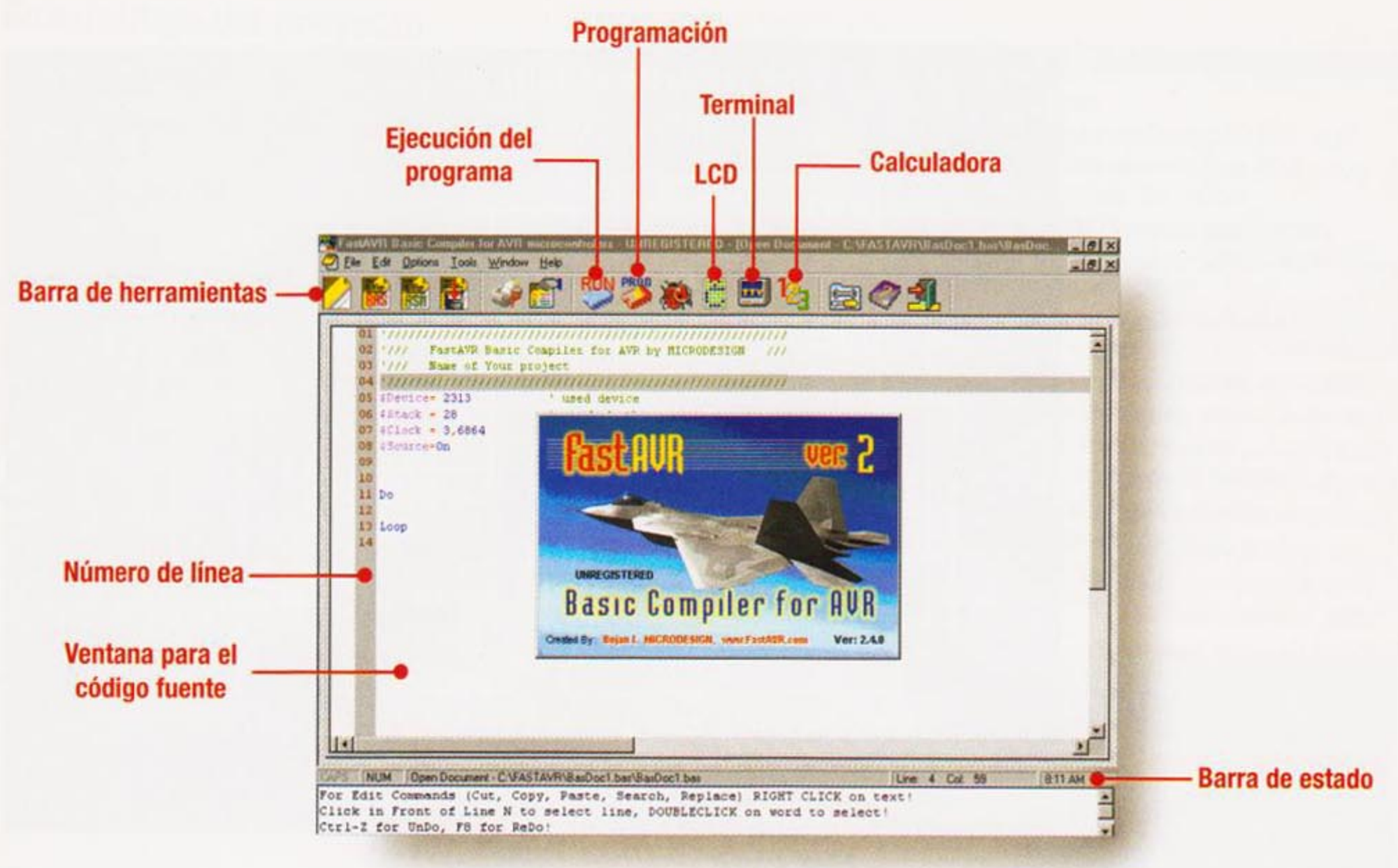


Figura 32.12 Entorno de compilación FASTAVR

El programa tiene una presentación visual bastante agradable aunque no es tan completo como el MPLAB de Microchip o el entorno para Motorola de Pemicro, sin embargo, posee algunas herramientas bastante interesantes.

En la **figura 32.13** vemos una de ellas. Se trata de un programa para trabajar con los puertos seriales, algo similar al *hyperterminal* de Windows 98; cuenta con su menú de configuración para la velocidad, la selección del puerto, etc.

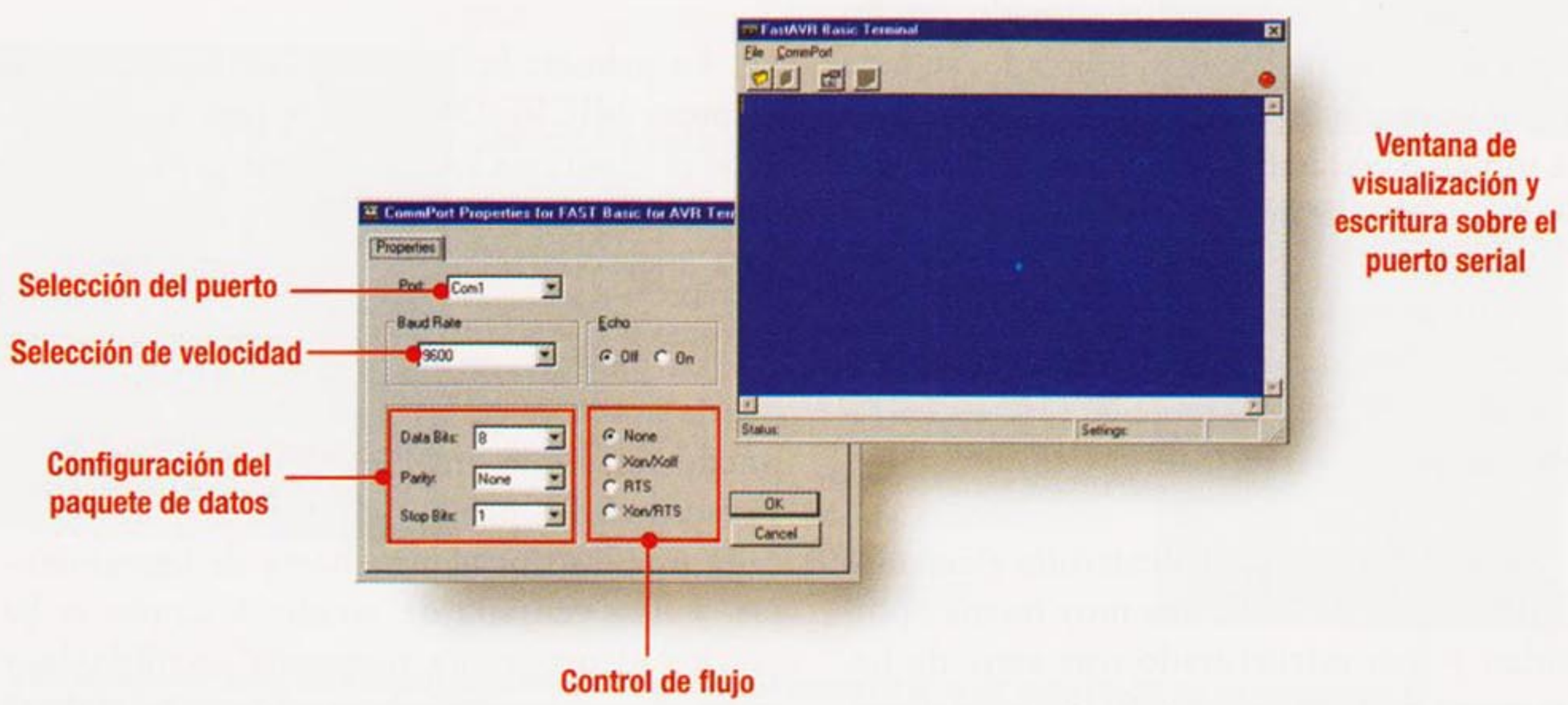


Figura 32.13 Programa para trabajar con los puertos

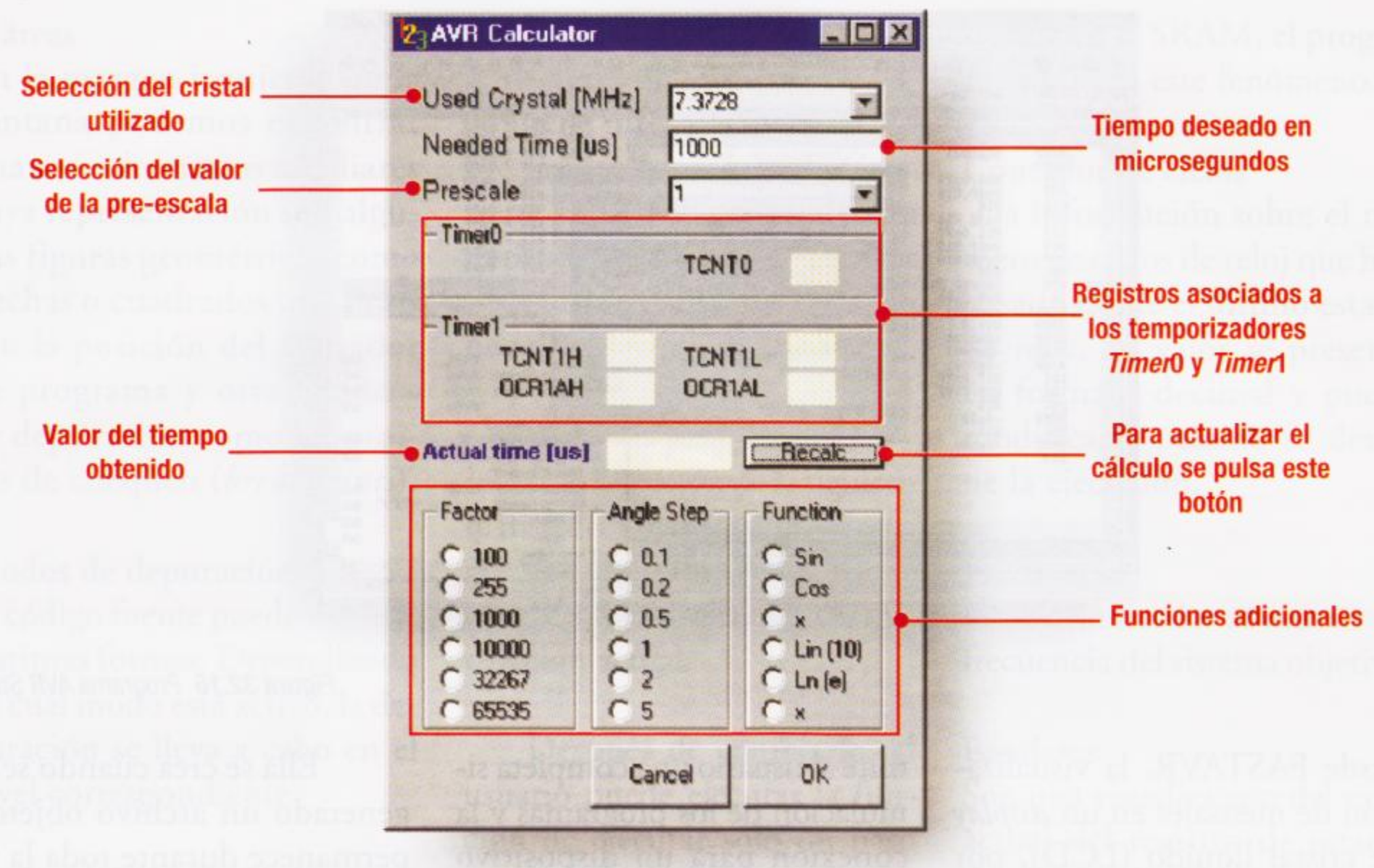


Figura 32.14 Calculadora para el cálculo de los temporizadores en los microcontroladores AVR

En la **figura 32.14** vemos otra herramienta bastante original y útil. Se trata de una calculadora para manipular los parámetros de los temporizadores con el objetivo de obtener un tiempo deseado con mucha precisión. Esta llamativa característica nos ahorra el trabajo de tener que utilizar una calculadora de mano y hacer nosotros mismos los cálculos del reloj, la pre-escala, etc.

Otra herramienta bastante original, que ha desarrollado MICRODESIGN, es la incorporación de una librería para generar rápidamente,

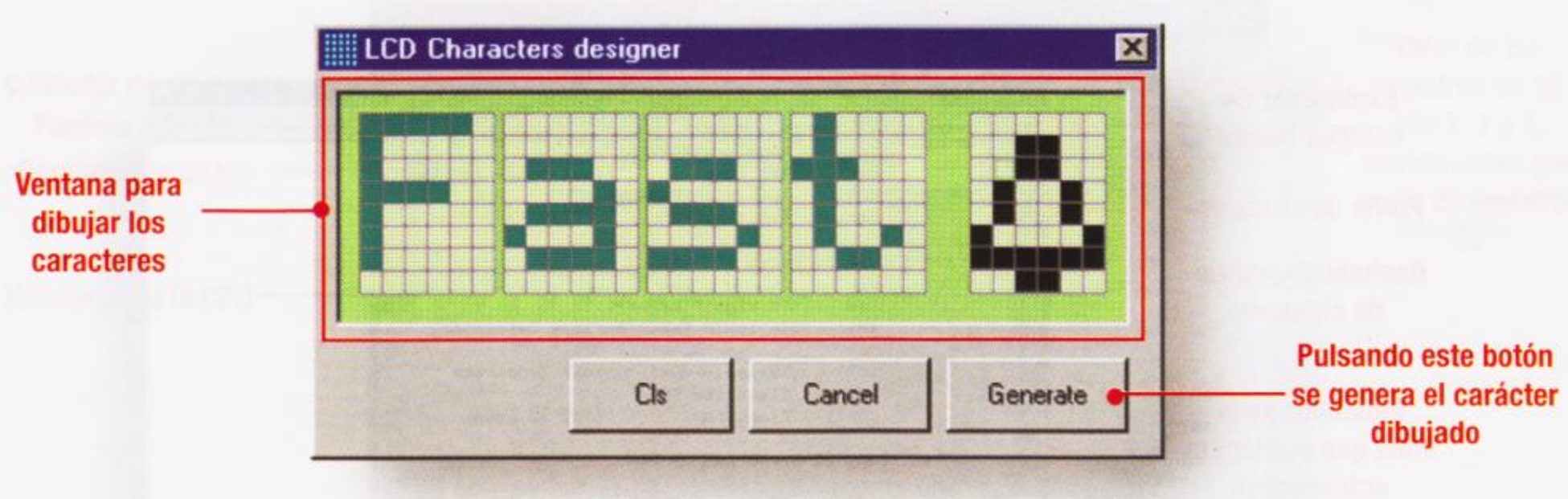


Figura 32.15 Interfaz para la generación de caracteres LCD.

Figura 32.16 *Programa AVR Studio*

desde FASTAVR, la visualización de mensajes en un *display* de cristal líquido (LCD), por medio de una sencilla interfaz gráfica, **figura 32.15**.

AVR Studio

La segunda herramienta que recomendamos se puede descargar de la página **www.atmel.com**, se trata del programa AVR Studio, **figura 32.16**. Este programa permite al usuario una completa simulación de los programas y la conexión para un dispositivo emulador. Observemos ahora sus principales componentes y su correspondiente función.

La ventana del código fuente

Es la ventana principal en cualquier sesión de trabajo o depuración, **figura 32.17**.

Ella se crea cuando se ha generado un archivo objeto y permanece durante toda la sesión. Si se cierra esta ventana, la sesión de depuración también se cierra. En ella se visualiza el código fuente con el que se está trabajando, ya sea en modo simulador o en modo emulador, y cuenta con una serie de indicadores que la hacen muy versátil y útil.

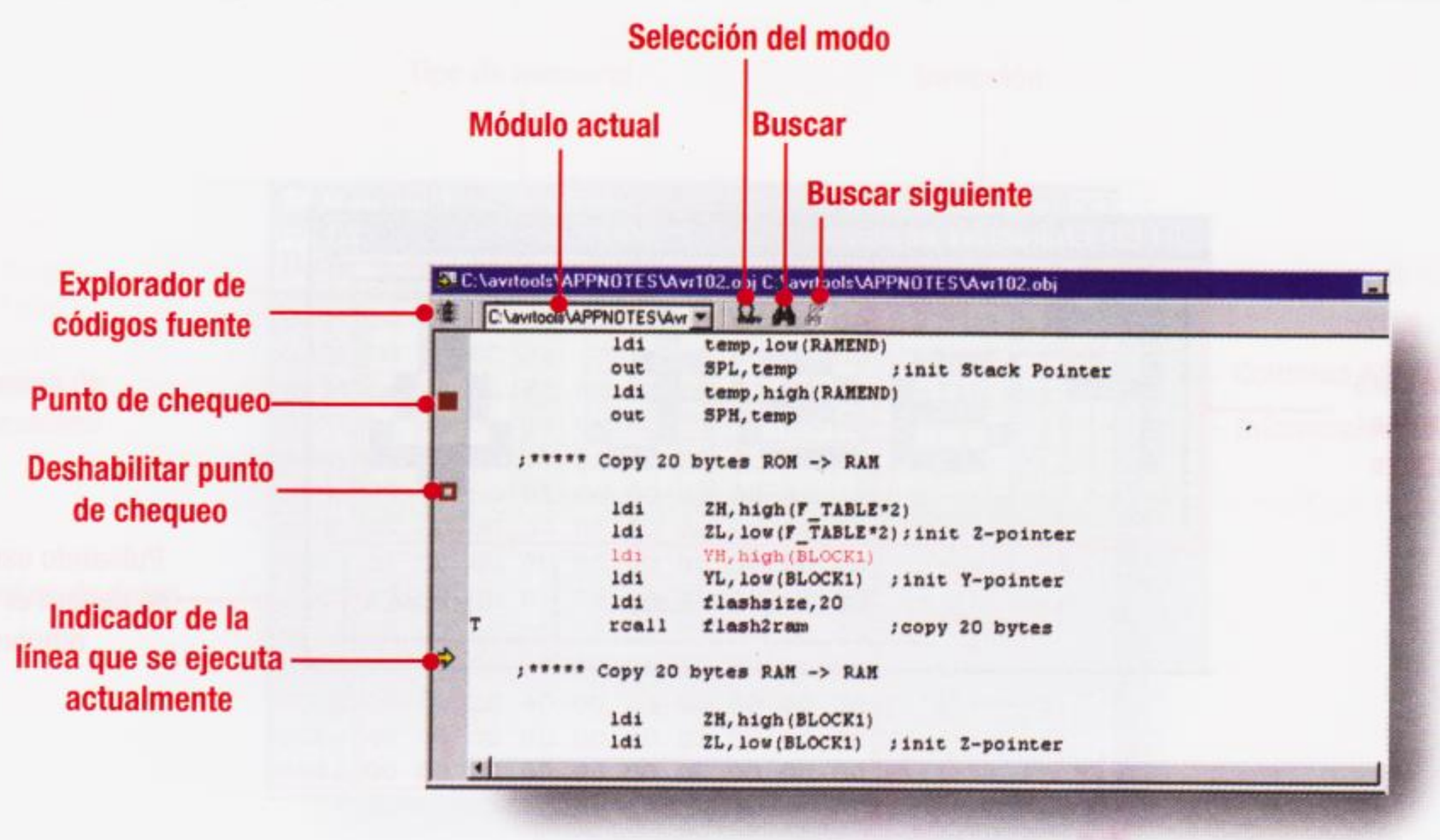


Figura 32.17 *Ventana del código fuente*

Marcas
En la margen izquierda de la ventana podemos encontrar una serie de marcas auxiliares cuya representación son algunas figuras geométricas como flechas o cuadrados que indican la posición del contador de programa y otros puntos de depuración como los puntos de chequeo (*breakpoints*).

Modos de depuración
El código fuente puede verse de distintas formas. Dependiendo de cual modo está activo, la depuración se lleva a cabo en el nivel correspondiente.

Explorando los módulos
Un archivo objeto puede tener varios módulos, pero sólo se puede visualizar uno en cada instante de tiempo. El usuario puede cambiar a otros módulos, según el que requiera, seleccionándolos desde un menú ubicado en la parte superior de esta ventana.

Ventana del procesador
Esta ventana, **figura 32.18**, contiene información vital sobre la ejecución actual. Contiene información vital para depurar adecuadamente los programas; resaltemos cada una de sus partes constitutivas.

Contador de programa
Indica la dirección de la siguiente instrucción que se va a ejecutar. Si se modifica el valor de este contador, no se ejecutará la instrucción actual.

Después de cambiarlo, el usuario puede ejecutar la función de ejecutar sólo un paso para saltar a la dirección deseada y no a la que debería seguir normalmente.

Puntero a la pila
Mantiene el valor actual del puntero a la pila, el cual se ubica en el área I/O. Si el sistema objetivo posee un *hardware* para la pila en lugar de una pila basada en la SRAM, el programa indicará este fenómeno.

Contador de ciclos
Da información sobre el número de ciclos de reloj que han pasado desde el último estado de *reset*. Su valor se presenta en formato decimal y puede modificarse cuando se detiene la ejecución.

Frecuencia
En este indicador se muestra la frecuencia del sistema objetivo.

Banderas
Son una visualización del valor actual del registro de estado. Cuando se detiene la ejecución, estos *bits* pueden modificarse pulsando el cursor sobre la bandera respectiva.

Ventana I/O
Se utiliza para inspeccionar y modificar los contenidos de los distintos registros I/O durante la ejecución del progra-

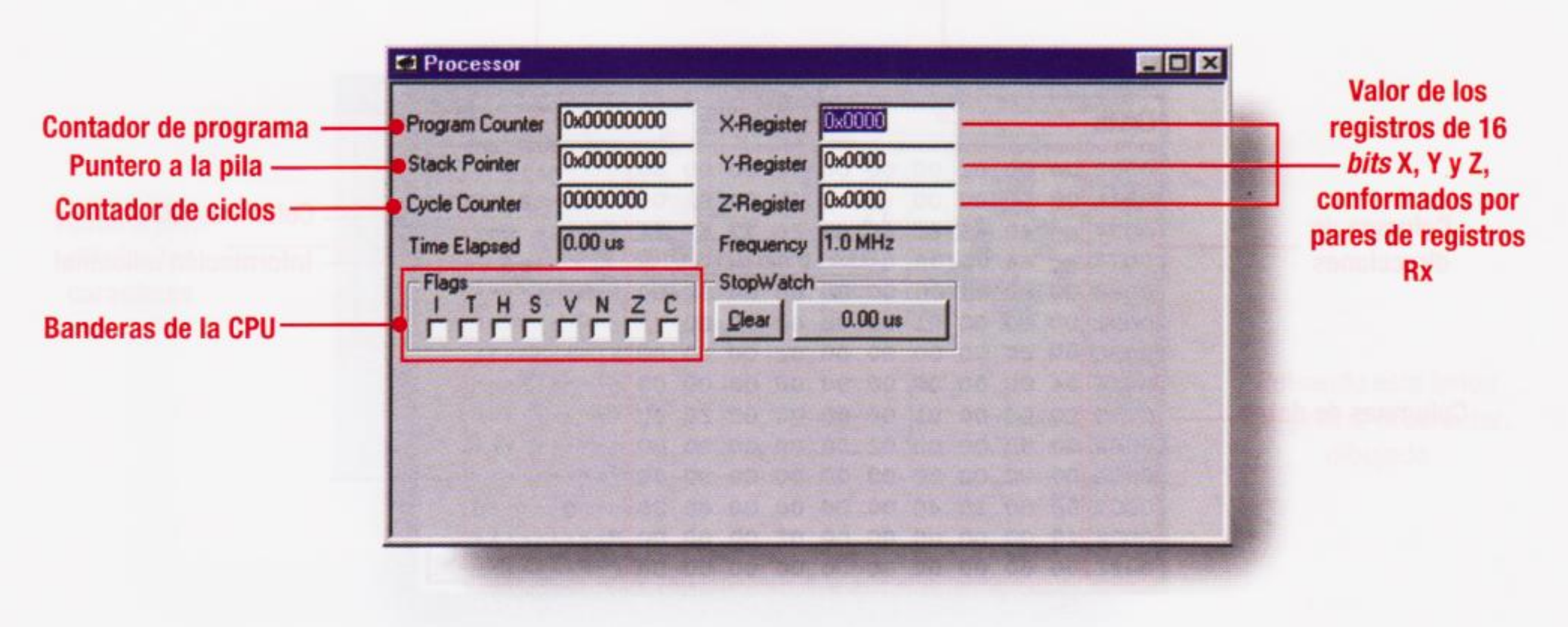


Figura 32.18 Ventana del procesador

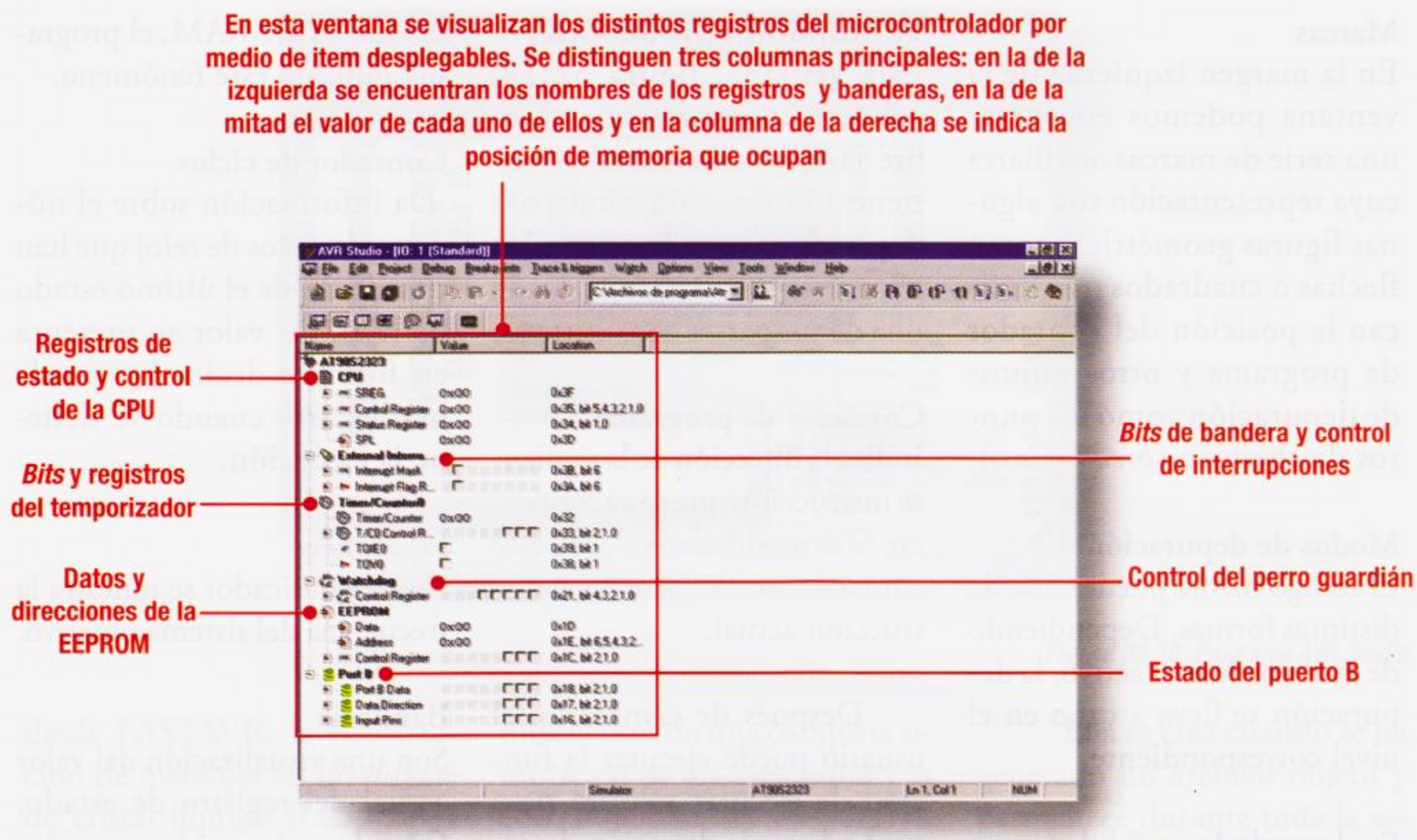


Figura 32.19 Ventana I/O

ma objetivo, **figura 32.19**. Se pueden abrir varias ventanas de este tipo seleccionando la opción **New I/O Window**, ubicada en el menú **View**.

Configuración de la ventana I/O

Esta ventana se configura automáticamente cuando se abre de acuerdo a unos parámetros estándar. La ventana se puede configurar cargando un archivo desde el disco.

Cuando se cierra un proyecto, AVR Studio guarda au-

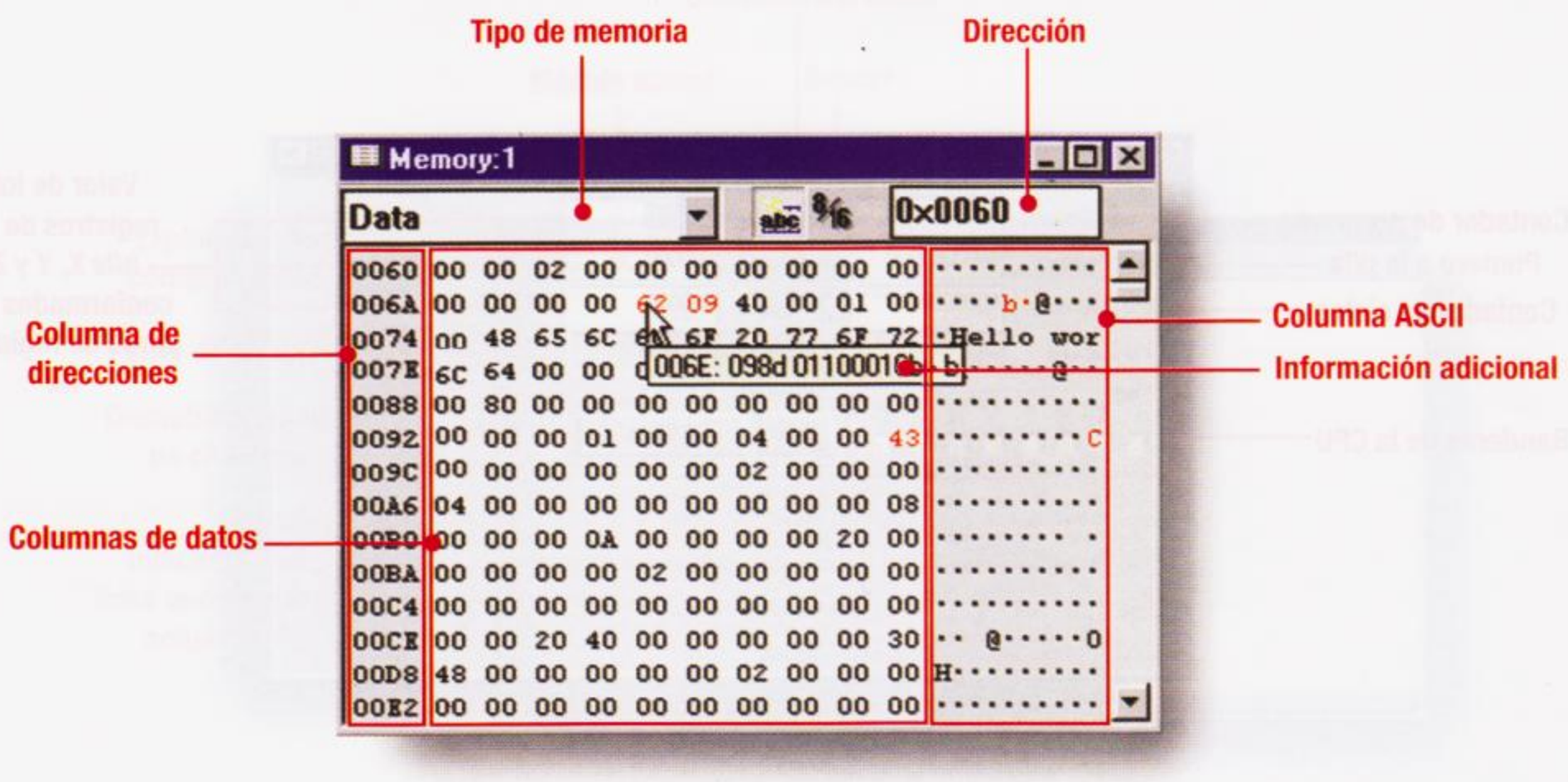


Figura 32.20 Ventana de memoria

tomáticamente los cambios hechos sobre dicha ventana.

Contenidos

La ventana I/O se organiza de manera jerárquica, lo que le da al usuario la posibilidad de ver los registros I/O, y, si lo desea, entrar en detalles.

La configuración estándar proporciona una rápida visión del *hardware* con la capacidad de expandir puntos en particular.

La ventana I/O contiene tres columnas. La columna de la izquierda muestra los registros; haciendo doble clic en cada uno de ellos, se expandirá el registro mostrando información adicional, incluyendo una pequeña pero muy eficaz gráfica de los registros y los *bits*.

La columna de la mitad muestra los valores de los registros y en la columna de la derecha se muestran las correspondientes direcciones.

Ventana de memoria

Le permite al usuario inspeccionar y modificar los contenidos de toda la memoria del sistema objetivo.

Se pueden abrir varias ventanas de este tipo, **figura 32.20**.

Información adicional

Ubicando el ratón sobre cualquier posición, se puede obtener más información sobre ella gracias a una pequeña ventana que se despliega automáticamente.

Ventana de variables

La ventana de observación (*Watch window*) permite visualizar los tipos y valores de las distintas variables de un programa, **figura 32.21**.

Sólo puede haber una ventana de observación abierta al tiempo. La ventana de observación contiene cuatro vistas, las cuales se seleccionan haciendo clic en las barras que se encuentran en su parte inferior.

Esta ventana consta de cuatro columnas. La primera es el nombre del símbolo que se quiere visualizar, la siguiente es el valor del símbolo, la tercera es el tipo del símbolo, y la cuarta es la dirección.

Por defecto, la ventana de observación se encuentra vacía y todos los símbolos y variables se deben adicionar de acuerdo a las necesidades del usuario.

Al incorporar un registro en esta ventana, podremos ver el comportamiento de cada uno de sus *bits* cuando depuramos paso a paso el programa.

Para abrir la ventana de observación se selecciona la opción **Watch**, presente en el menú **View** o se hace clic sobre **Watch Window** en la barra de herramientas **Views**.

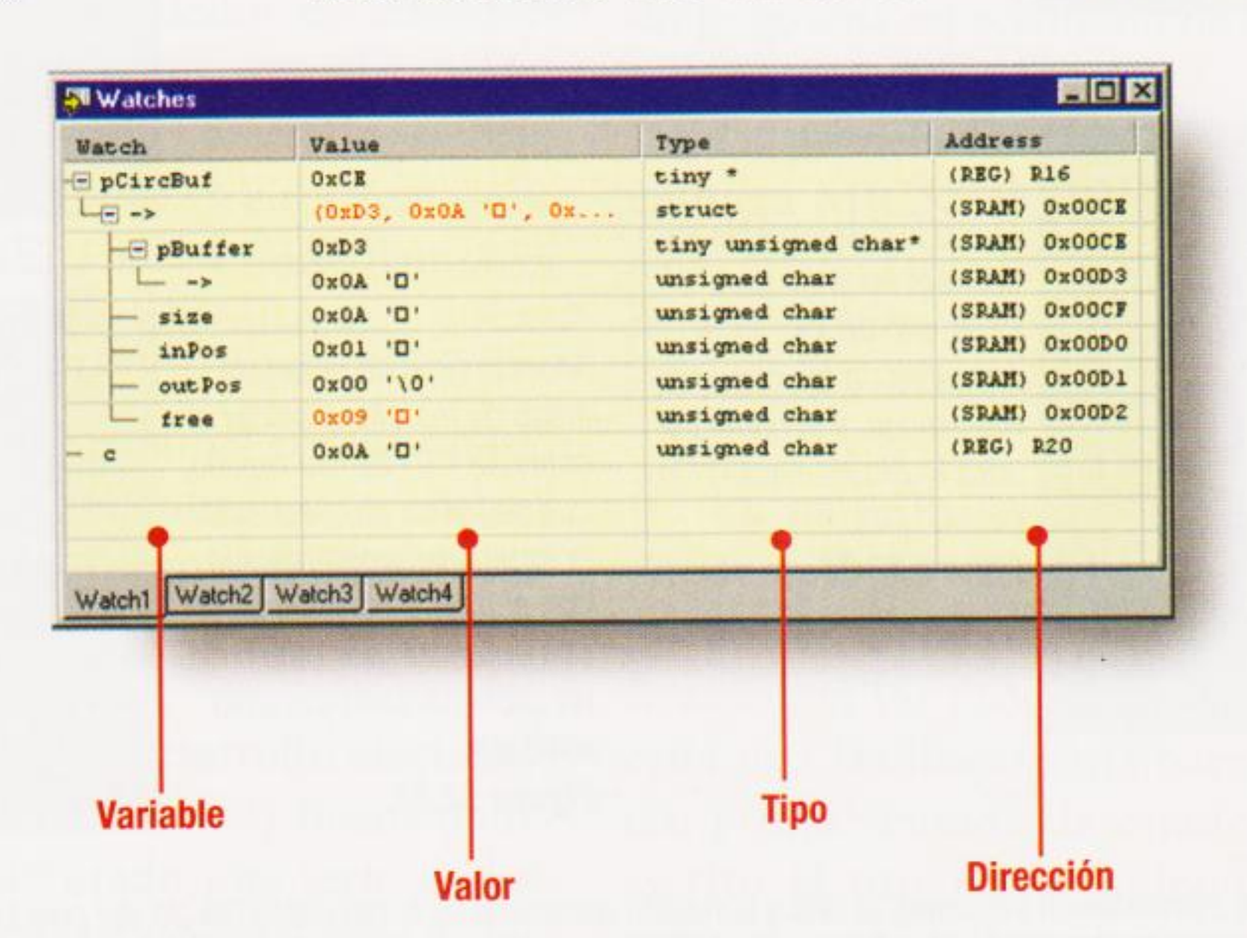


Figura 32.21 Ventana de observación

Ensamblaje del proyecto

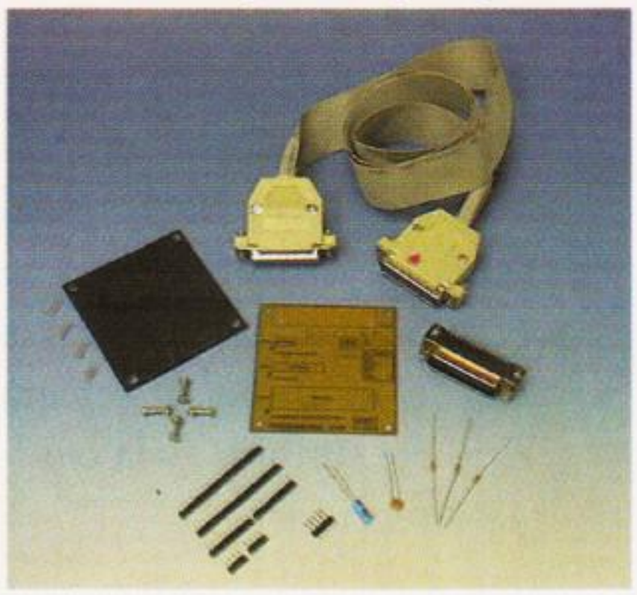


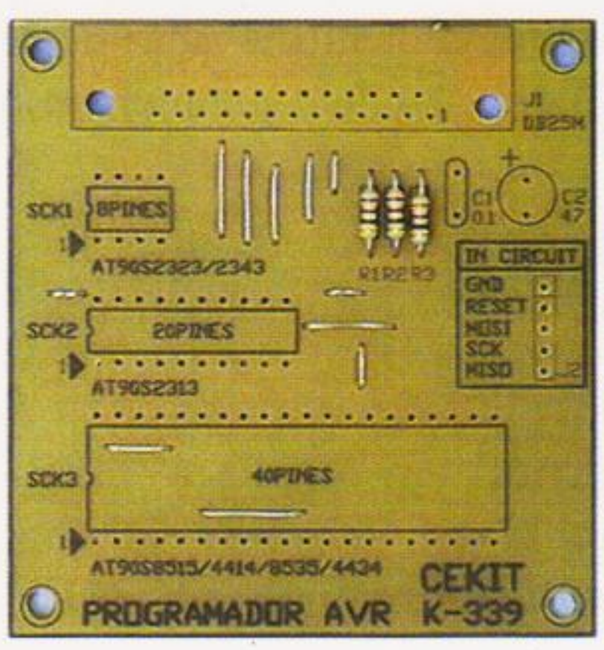
Figura 32.22 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el programador de microcontroladores ATMEL, CEKIT K-339.*

Con este programador usted podrá grabar los microcontroladores más populares de ATMEL, utilizando el puerto paralelo de la computadora. En este proyecto se han incluido sockets que facilitan la programación de los circuitos integrados de 8, 20 y 40 pines. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

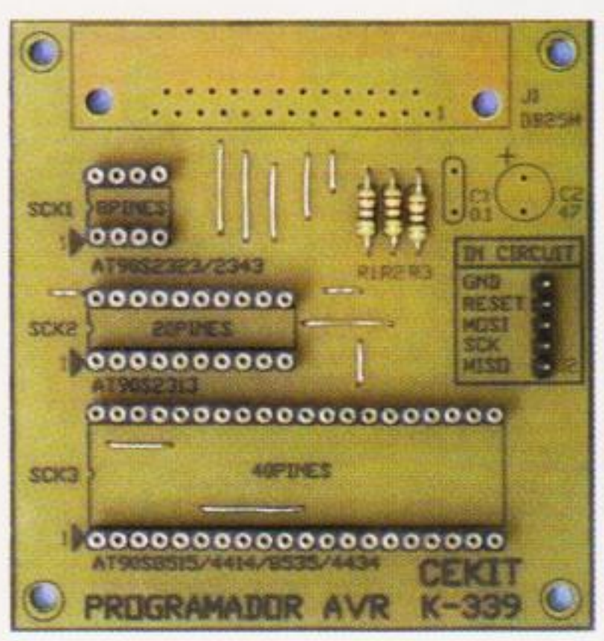
-

LISTA DE MATERIALES

Cant.	Referencia	Descripción
1	C1	Condensador cerámico de 0,1uF - 50V
1	C2	Condensador electrolítico de 47uF - 16V
3	R1 a R3	Resistencias de 220 - 1/4W
1	J1	Conector DB25 macho para impreso
1	J2	Conector tipo cerca no polarizado de 5 pines
1	K-339	Circuito impreso CEKIT
1		Cable para computadora paralelo-paralelo (hembra - macho)
2		*Sockets* en línea de 20 pines
2		*Sockets* en línea de 10 pines
2		*Sockets* en línea de 4 pines
1		Lámina de acrílico de 8cm X 8cm
4		Separadores plásticos de 5mm
4		Tornillos milimétricos 3X12
4		Tuercas de 3mm

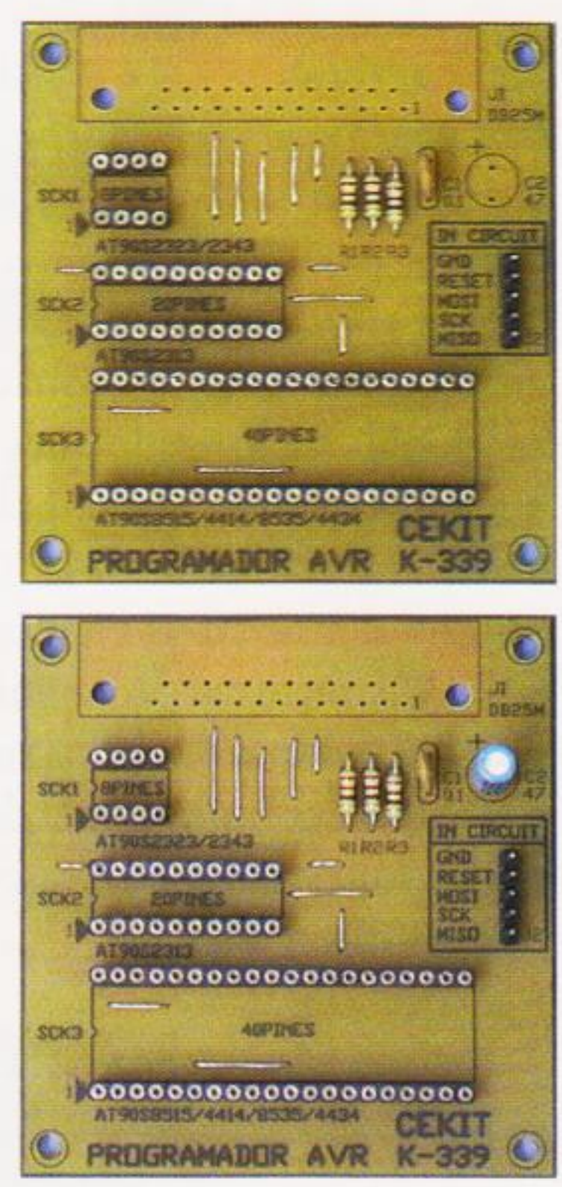


Paso 1
Instale y suelde los puentes de alambre y las tres resistencias de 220Ω.
Figura 32.23



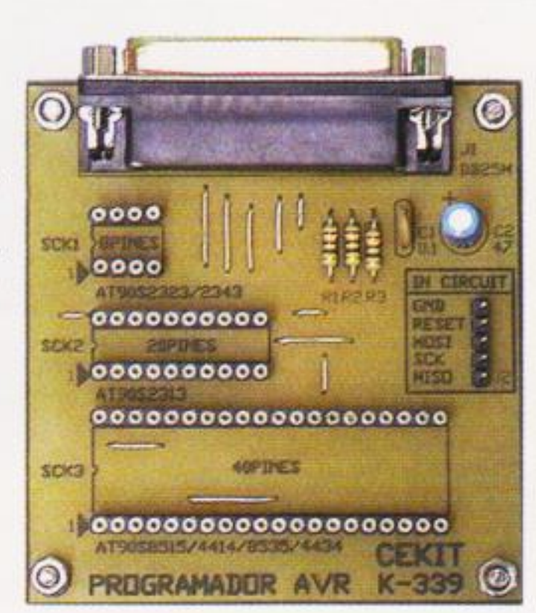
Paso 2
Luego instale y suelde los sockets en línea para cada uno de los formatos de los circuitos integrados, y el conector para la programación "in circuit" (J2).
Figura 32.24

Paso 3
Instale y suelde el condensador cerámico C1 de 0,1µF.
Figura 32.25



Paso 4
Instale y suelde el condensador electrolítico C2 de 47µF.
Figura 32.26

Paso 5
Aspecto final de la tarjeta luego de instalar y soldar el conector (J1) DB-25 macho, que se utiliza para conectar la tarjeta de programación con el puerto paralelo de la computadora, por medio de un cable paralelo-paralelo estándar.
Figura 32.27



Paso 6. Pruebas finales
Esta tarjeta no necesita alimentación adicional, la única conexión necesaria es la del cable DB-25 del puerto paralelo. El programa funciona de manera inmediata; si se presenta algún incoveniente en la programación cambie el microcontrolador, si persisten los problemas es posible que el puerto paralelo presente algún daño.

PROYECTO 33

MEDIDOR DE HUMEDAD Y TEMPERATURA

En este proyecto construiremos un dispositivo de medida que muestra en un *display* de cristal líquido (LCD) dos variables análogas, cuya medición es una necesidad permanente en los procesos industriales; se trata de la temperatura en grados centígrados y la humedad relativa medida en porcentaje.

Características

- Visualización de la temperatura y la humedad relativa en un *display* LCD de dos líneas y dieciséis caracteres (2x16)
- Sensor de humedad capacitivo
- Sensor en circuito integrado para la temperatura
- Intervalo de medición de temperatura entre 0 y 150º C
- Microcontrolador PIC16C711

Funcionamiento

El núcleo del proyecto es un microcontrolador PIC16C711 que representa una muy buena solución para muchos problemas de diseño donde se requiera trabajar con señales análogas y digitales. En el interior de este microcontrolador PIC existe un convertidor análogo/digital cuyas principales características son:

- Cuatro canales de entrada
- Tiempo de conversión mínimo de 20 us
- Voltaje de referencia interno o externo
- Resolución de 8 *bits*
- Magnitud de la entrada análoga desde Vss hasta Vref

Como característica importante podemos mencionar que una de las presentaciones con la que se consigue en el mercado es la de montaje superficial, que lo hace muy conveniente para incorporarlo en aparatos de tamaño reducido ,y, por su bajo costo, se convierte en la clave del éxito para producciones en serie.

El PIC16C711 se encarga de adquirir las dos señales (previamente acondicionadas) de los sensores de humedad y temperatura, y de generar las señales de control para poder visualizar los datos tomados en las dos líneas del *display* LCD. Básicamente ese es el funcionamiento y el objetivo perseguido en este proyecto; a pesar de parecer simple, esta aplicación de los microcontroladores es muy apetecida (por ejemplo en los invernaderos) por brindar independencia de una computadora para tomar una lectura y por la estética que brinda un LCD.

Descripción del circuito

Podemos distinguir tres bloques fundamentales en este proyecto: los sensores, la etapa de proceso y la etapa de visualización de los datos, **figura 33.1**. Analicemos brevemente cada una de las etapas y los circuitos que las componen; el diagrama completo del proyecto con todos los componentes y sus conexiones, lo podemos apreciar en la **figura 33.2**.

Figura 33.1 *Diagrama en bloques del proyecto*

Sensores

Para medir la temperatura emplearemos un circuito integrado LM35. Este sensor tiene tres pines: alimentación, tierra y la salida análoga. Este dispositivo presenta en su salida una variación de 10 mV/°C, por lo tanto, el valor de la temperatura se puede recibir directamente, sin necesidad de hacer modificaciones al dato obtenido.

El LM35 puede trabajar en una escala de temperatura entre –55 y 150°C, y la fuente de alimentación positiva puede estar entre 4 y 30 voltios. Además, su precisión es de 0,5°C. Este elemento viene en un encapsulado plástico y tiene la misma apariencia de un transistor.

Para medir la humedad utilizaremos un sensor capacitivo de Philips, **figura 33.3**, cuyo comportamiento responde a la gráfica mostrada en la **figura 33.4**. Este sensor se caracteriza por su facilidad de manejo y su bajo costo; y su diseño permite medir valores entre 10% y 90% de humedad relativa. Como vemos en la gráfica de la **figura 33.4**, el sensor presenta una respuesta bastante lineal, lo que permite ser leído mediante circuitos electrónicos no muy complicados. Básicamente su funcionamiento se basa en una fina membrana polímera capaz de absorber la humedad que hay a su alrededor originando un cambio en el valor de su capacitancia. Señalemos las principales características del sensor de humedad:

- Larga vida útil
- Alta sensibilidad
- Respuesta rápida
- Alta inmunidad contra contaminantes
- Opera con un sencillo circuito de medida

Dentro de las aplicaciones más comunes, en donde se puede encontrar este sensor, se encuentran los procesos de secado, sistemas de

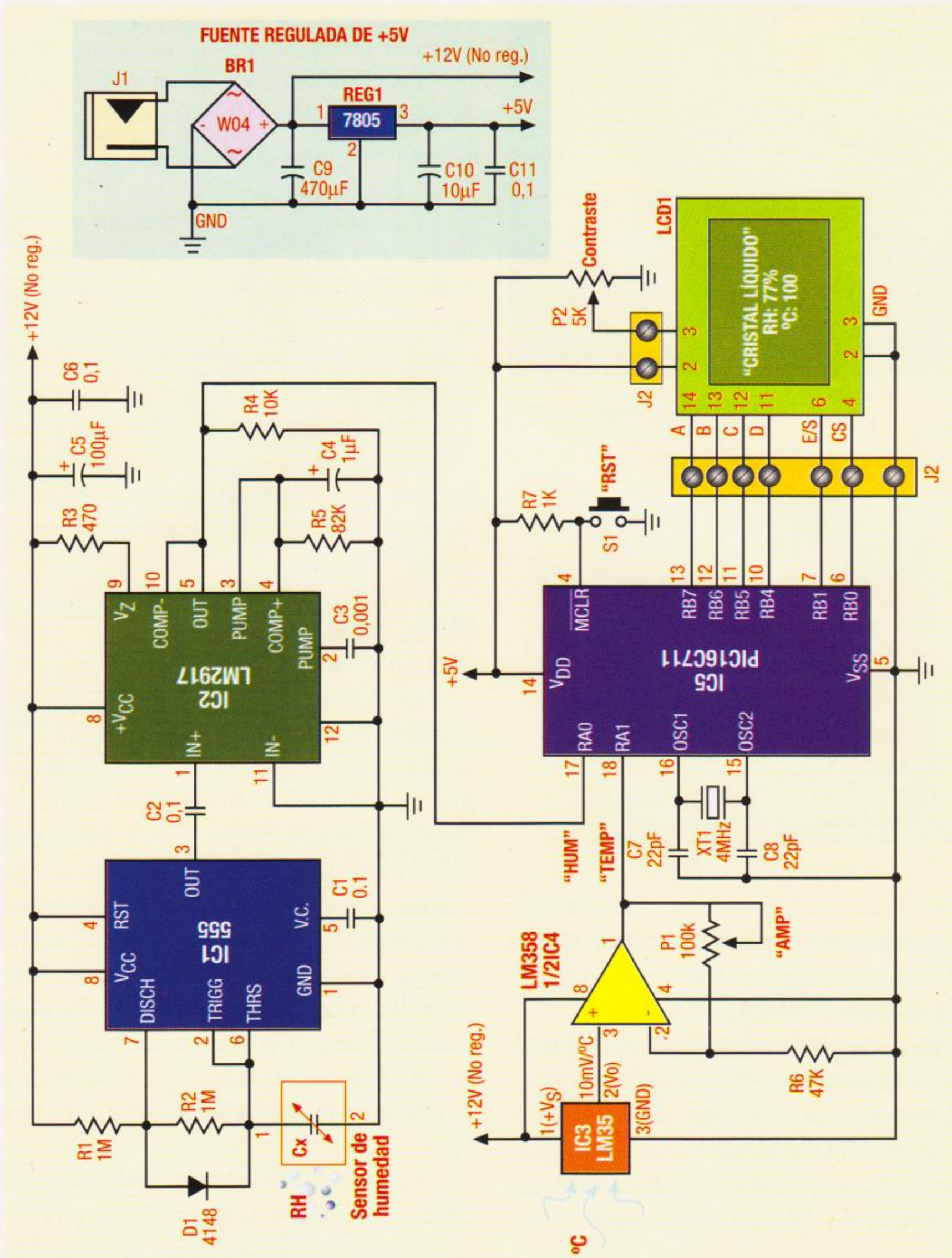


Figura 33.2 *Diagrama esquemático del proyecto*

aire acondicionado, sistemas de impresión, estaciones meteorológicas, etc. Para originar la señal análoga, el sensor se instala en un circuito 555 configurado en modo astable para que envíe una frecuencia dependiendo del valor de la capacitancia que se tenga en cada momento; con los componentes utilizados la frecuencia de esa señal puede estar entre 5 y 6 KHz. Por medio de un circuito integrado convertidor de frecuencia a voltaje, referencia LM2917 de National Semiconductor, obtenemos una señal análoga entre 2 y 5 voltios proporcional a la humedad y lista para ser leída por el microcontrolador.

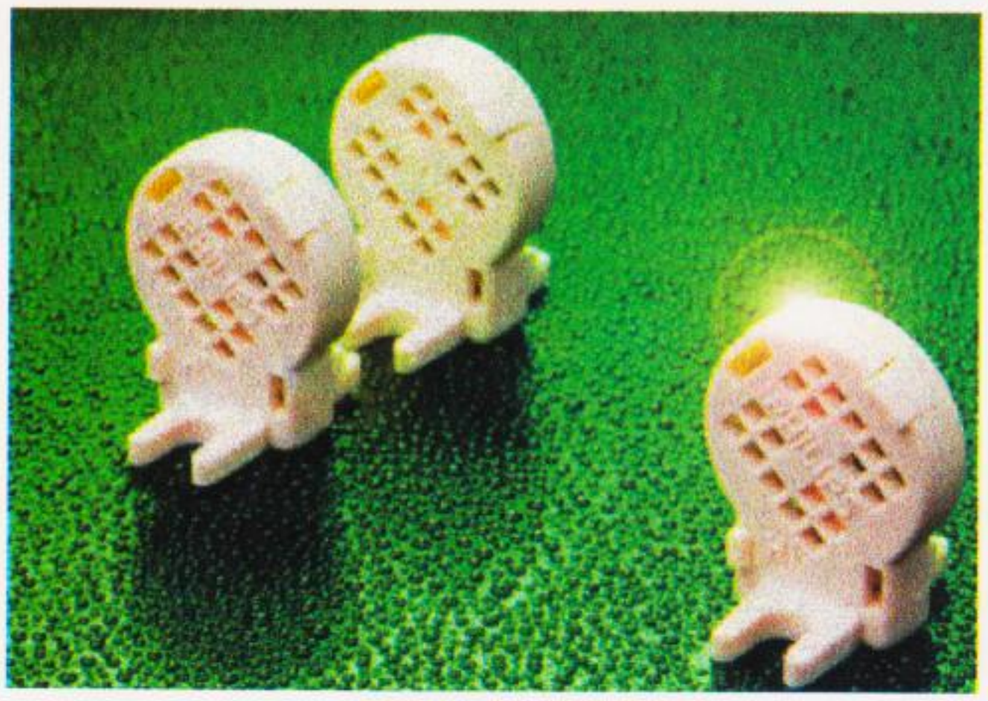


Figura 33.3 Sensor de humedad capacitivo

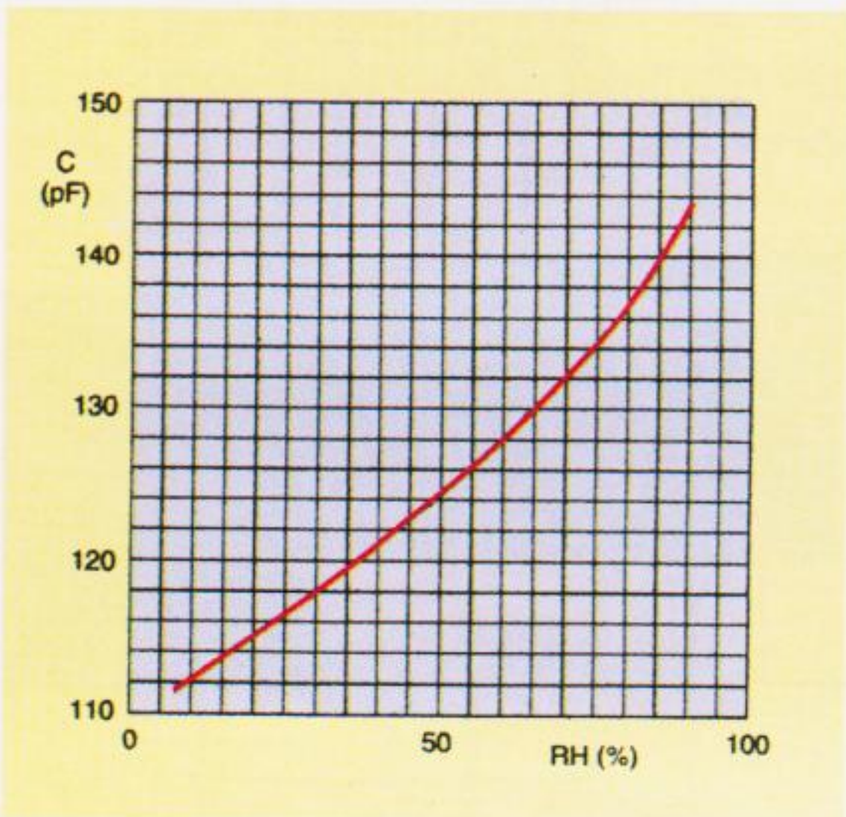


Figura 33.4 Comportamiento del sensor: capacitancia contra humedad

Etapa de procesamiento

Las entradas análogas se conectan a los pines RA0 (canal 0 del convertidor) y RA1 (canal 1) que corresponden a los pines 17 y 18 del microcontrolador PIC 16C711. El voltaje de referencia del convertidor se configura para que corresponda al voltaje de alimentación del microcontrolador.

La máxima lectura del convertidor análogo a digital (0FFH) se obtiene cuando la entrada análoga sea igual o superior a 5 voltios. De tal manera, que si hacemos la división de 5 voltios entre 255 (FFH), obtenemos una relación de aproximadamente 20 mV por unidad. Por ejemplo en el caso de la temperatura, si la entrada análoga está en 160 mV (es decir 16 °C), el valor digital que entrega el convertidor es 8 (160/20).

Dado lo anterior, el valor que entrega el convertidor se debe duplicar para compensar la diferencia (variación de 10 mV/°C y resolución de lectura de 20 mV), aunque esto hace que la lectura siempre sea un número par.

Para solucionar el problema de tener que duplicar el valor leído y garantizar que el resultado de la conversión corresponda a la temperatura real, se puede utilizar un amplificador operacional para amplificar la señal entregada por el sensor LM35; dicho operacional se debe configurar como amplificador no inversor. De esta forma, con el valor análogo adecuado a la entrada del convertidor y con la fuente de alimentación como voltaje de referencia, se puede obtener una lectura precisa y directa.

Etapa de visualización

El módulo de cristal líquido se conecta al microcontrolador utilizando una interfaz de cuatro *bits*. El manejo de los *displays* LCD ya lo estudiamos anteriormente en la lección de programación, pero vamos a hacer una síntesis de los aspectos más importantes. Ellos están dotados de toda la electrónica necesaria para visualizar datos de manera muy eficiente. Además disponen de un buen conjunto de instrucciones, con elementos tales como borrar la pantalla, posicionar el cursor, desplazar los mensajes, etc., las cuales satisfacen la mayor parte de las necesidades de los usuarios para el despliegue de mensajes.

La arquitectura de un módulo LCD la podemos apreciar en la **figura 33.5**; en ella podemos distinguir la memoria RAM la cual está destinada a almacenar la información de los mensajes, éstos pueden estar en: área visible si están desde la posición 00H hasta la 0FH para la primera línea, ó desde la 40H hasta la 4FH para la segunda línea, **figura 33.6**. Otro bloque es la memoria ROM, en ella se encuentra grabada la información necesaria de los puntos para crear los caracteres alfanuméricos y símbolos (ASCII).

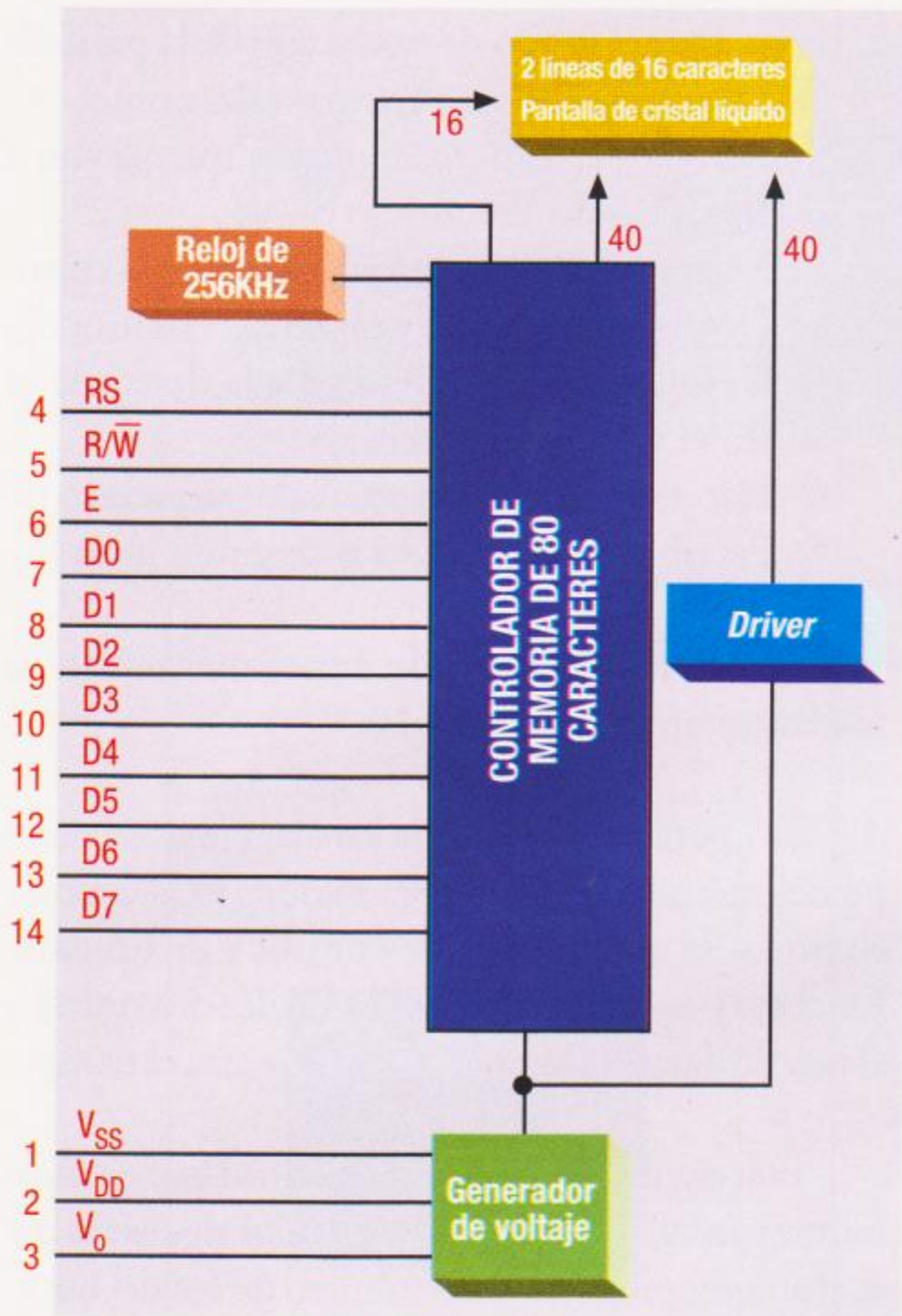


Figura 33.5 *Diagrama de bloques simplificado de la arquitectura de un módulo LCD*

Los terminales de conexión pueden variar un poco entre los distintos modelos; en la **figura 33.7** vemos una distribución típica de ellos. La alimentación del módulo se indica en la **figura 33.8**, el terminal Vo del módulo se obtiene de la derivación del potenciómetro. El voltaje de la fuente de alimentación y su correcta conexión son críticos; no se permiten errores es este sentido ya que pueden resultar fatales para el LCD.

PRINCIPALES CARACTERÍSTICAS

Dimensiones	84mm X 44mm X 12mm
Área visible	61,0mm X 15,8mm
Formato de los caracteres	Matriz de 5x7 puntos
Peso	25g*
Alimentación	+5 voltios
Consumo	7,5mW (equiv. I=1,5mA)
Bus de transferencia de datos	8 ó 4 bits
Número de terminales de conexión	14
Tiempo de encendido típico a 25ºC	110ms
Tiempo de apagado típico a 25ºC	110ms

Para manejar un LCD se requiere de un sistema digital basado en microprocesador, microcontrolador o computadora, **figura 33.9**. Recordemos los aspectos fundamentales para el manejo del LCD:

1. Interconexión apropiada con el sistema digital

Ésta se consigue interconectando el módulo LCD y el sistema digital a través del bus de datos (4 ó 8 líneas) y las señales de control (3 líneas). El control necesita las señales RS, R/$\overline{W}$ y E. RS para comunicar si el dato en el bus es un carácter ASCII o una palabra de control, R/$\overline{W}$ para informar si se quiere leer o escribir en el módulo y E, para habilitar la operación.

2. Programar el número de bits para el bus de datos

Esta programación puede ser en ocho o en cuatro *bits*, la primera "entra" por defecto (al encender la fuente), esto quiere decir que no necesita programarse. La segunda, en cambio, requiere de una secuencia cuidadosa recomendada por los fabricantes; pero vale la pena, teniendo en cuenta las cuatro líneas que nos podemos economizar.

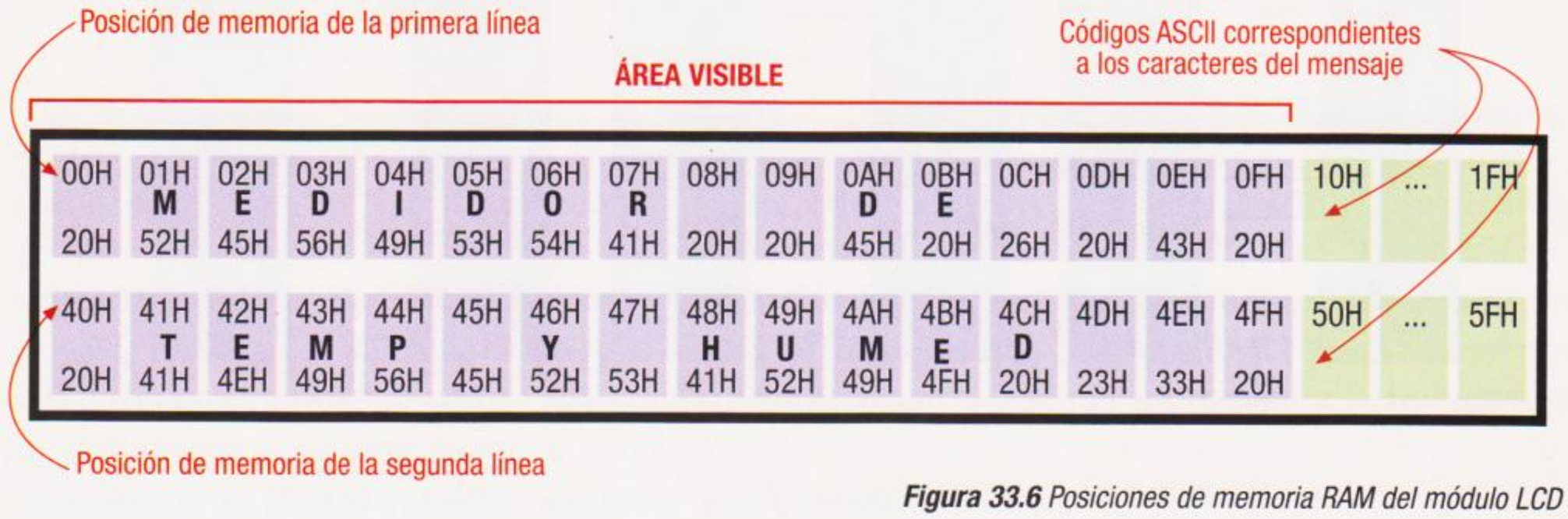


Figura 33.6 *Posiciones de memoria RAM del módulo LCD*

TERMINAL	SÍMBOLO	NOMBRE Y FUNCIÓN
1	V_{SS}	Tierra, 0V
2	V_{DD}	Alimentación +5V
3	Vo	Ajuste de voltaje/contraste
4	RS	Selección control/dato
5	R/$\overline{W}$	Lectura/escritura en LCD
6	E	Habilitación
7	D0	*Bit* menos significativo dato 2^0
8	D1	*Bit* dato 2^1
9	D2	*Bit* dato 2^2
10	D3	*Bit* dato 2^3
11	D4	*Bit* dato 2^4
12	D5	*Bit* dato 2^5
13	D6	*Bit* dato 2^6
14	D7	*Bit* más significativo 2^7

Figura 33.7 *Terminales de conexión del módulo LCD*

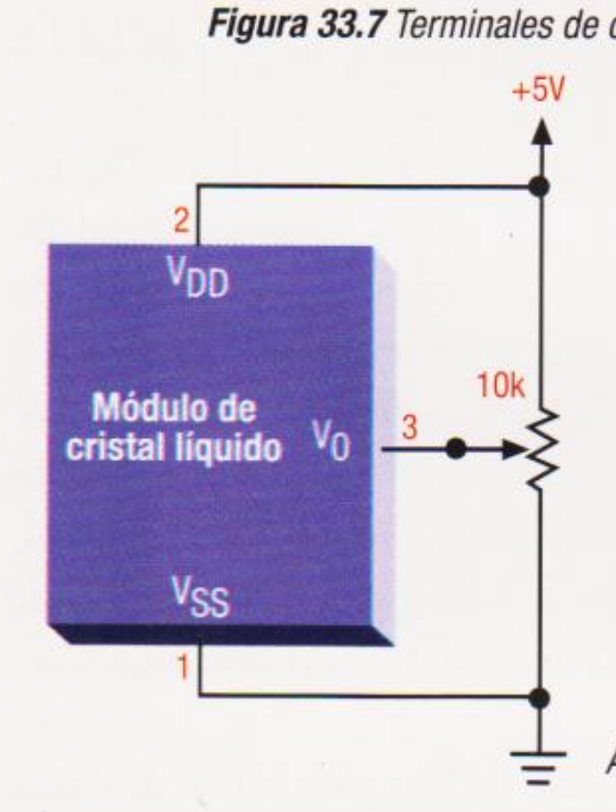


Figura 33.8 *Alimentación y contraste*

3. Programar el modo de operación de la pantalla

Éste puede incluir elementos tales como:

- Activar el número de líneas que se van a visualizar en el módulo
- Mantener el mensaje fijo y desplazar el cursor
- Desplazar el mensaje y mantener el cursor fijo
- Desplazar a la izquierda o a la derecha, ya sea el cursor o el mensaje
- Hacer que el carácter señalado parpadee o no
- Escribir en la primera o segunda línea

Todos los códigos de funcionamiento se muestran en la **figura 33.10**.

La **figura 33.11** muestra los diagramas de tiempo del módulo LCD, en ella se observa el ciclo de escritura de una **palabra de control** y de un **dato**. En el eje **y** están localizadas las señales de control y el bus de datos, y en el eje **x** se encuentra el tiempo.

Para escribir una palabra de control, por ejemplo **borrar pantalla**, es necesario seguir los siguientes pasos (una vez programado el número de *bits* del bus):

- Colocar RS en bajo, porque se trata de una palabra de control

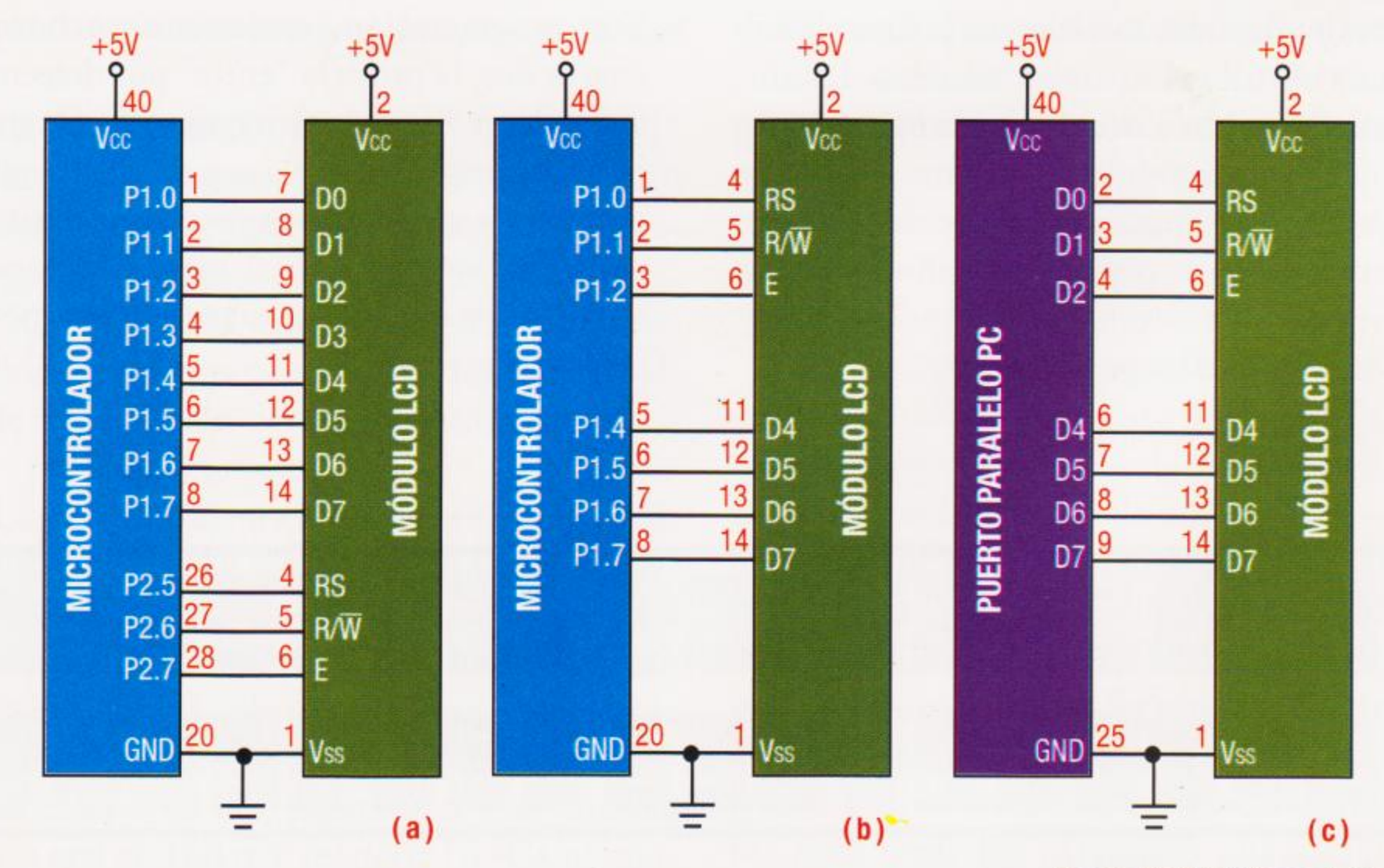


Figura 33.9 *Interfaces del módulo LCD*
a) *Con microcontrolador a 8 bits,* ***b)*** *Con microcontrolador a 4 bits,* ***c)*** *Con una computadora*

SIGNIFICADO DE LAS ABREVIATURAS

I/D = 1 Incrementa
= 0 Decrementa
S = 1 Desplaza el mensaje en la pantalla
= 0 Mensaje fijo en la pantalla
D = 1 Encender (activar) la pantalla
= 0 Apagar la pantalla (desactivar)

C = 1 Activar cursor
= 0 Desactivar cursor
B = 1 Parpadea carácter señalado por el cursor
= 0 No parpadea el carácter
S/C = 1 Desplaza pantalla
= 0 Mueve cursor

RL = 1 Desplazamiento a la derecha
= 0 Desplazamiento a izquierda
DL = 1 Datos de ocho bits
= 0 Datos de cuatro bits
BF = 1 Durante operación interna del módulo
= 0 Finalizada la operación interna

- Colocar R/$\overline{W}$ en bajo, porque se trata de un ciclo de escritura
- Colocar E en alto para preparar la habilitación
- Colocar el dato 00000001 binario en el bus y mantenerlo hasta cuando se habilite para escribir
- Colocar E en bajo. Este paso es el que finalmente borra la pantalla que era lo que se pretendía

Descripción del programa

En el programa se debe desarrollar la interfaz a 4 *bits* con el *display* de cristal líquido, una sección que se encarga de leer el convertidor análogo a digital, y una sección que convierte el dato hexadecimal obtenido de la conversión, a su equivalente en decimal, para ser mostrado en la pantalla.

Pasos que se deben seguir para hacer una conversión A/D:

1. Configure los pines del puerto A; los que vayan a trabajar como entradas análogas deben ser configurados como entradas en el registro TRISA.
2. Configure las entradas análogas y el voltaje de referencia (interno o externo) en el registro ADCON1.
3. Seleccione el canal de entrada en el registro ADCON0.
4. Seleccione el reloj de la conversión en ADCON0.
5. Encienda el convertidor (*bit* ADON del registro ADCON0) para que se tome la muestra de la señal análoga
6. Si se va a utilizar la interrupción del convertidor se debe borrar el *bit* ADIF del registro ADCON0 y poner "1" en el *bit* ADIE y el *bit* GIE del registro INTCON.

Control y dato / INSTRUCCIONES	Señal de control		DATO/DIRECCION								DESCRIPCION
	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	
Borrar pantalla	0	0	0	0	0	0	0	0	0	1	Limpia todo el display y retorna el cursor a la posición de inicio (dirección 0)
Cursor a casa	0	0	0	0	0	0	0	0	1	*	Retorna el cursor a la posición de inicio (dirección 0). También retorna el display, desplazándolo a la posición original. Los contenidos de la RAM DD permanecen sin cambio.
Seleccionar modo	0	0	0	0	0	0	0	1	1/D	0	Configura la dirección de movimiento del cursor y si se desplaza o no el display. Esta operación es realizada durante operaciones de lectura y escritura.
Encender/apagar pantalla	0	0	0	0	0	0	1	D	C	B	Configura el estado ON/OFF de todo el display (D), el cursor (C) y el parpadeo del carácter en la posición del cursor.
Desplazar Cursor/Pantalla	0	0	0	0	0	1	S/C	R/L	*	*	Mueve el cursor y desplaza el display sin cambiar los contenidos de la RAM DD.
Activar función	0	0	0	0	1	D/L	N	F	*	*	Configura el tamaño de la interfaz (DL), el número de líneas del display (N) y la fuente del carácter (F). N=0, 1 línea N=1, 2 líneas
CG RAM	0	0	0	1	Dirección generador de RAM						Ajusta la dirección del generador de caracteres. El dato CG RAM es enviado y recibido después de este ajuste.
DD RAM	0	0	1	Direcciónes de datos RAM							Ajusta la dirección de la RAM DD. La dirección es enviada y recibida después de este ajuste.
Bandera de ocupado	0	0	BF	AC							Lectura de la bandera Busy Flag, indicando que operaciones internas son realizadas, y lectura de los contenidos del contador de direcciones.
Escritura CG RAM/DD RAM	1	0	Escritura de Dato								Escribe datos en la RAM DD o en la RAM CG.
LECTURA CG RAM/DD RAM	1	1	Lectura de Dato								Lectura de datos desde la RAM DD o la RAM CG

Figura 33.10 *Conjunto de instrucciones del módulo LCD*

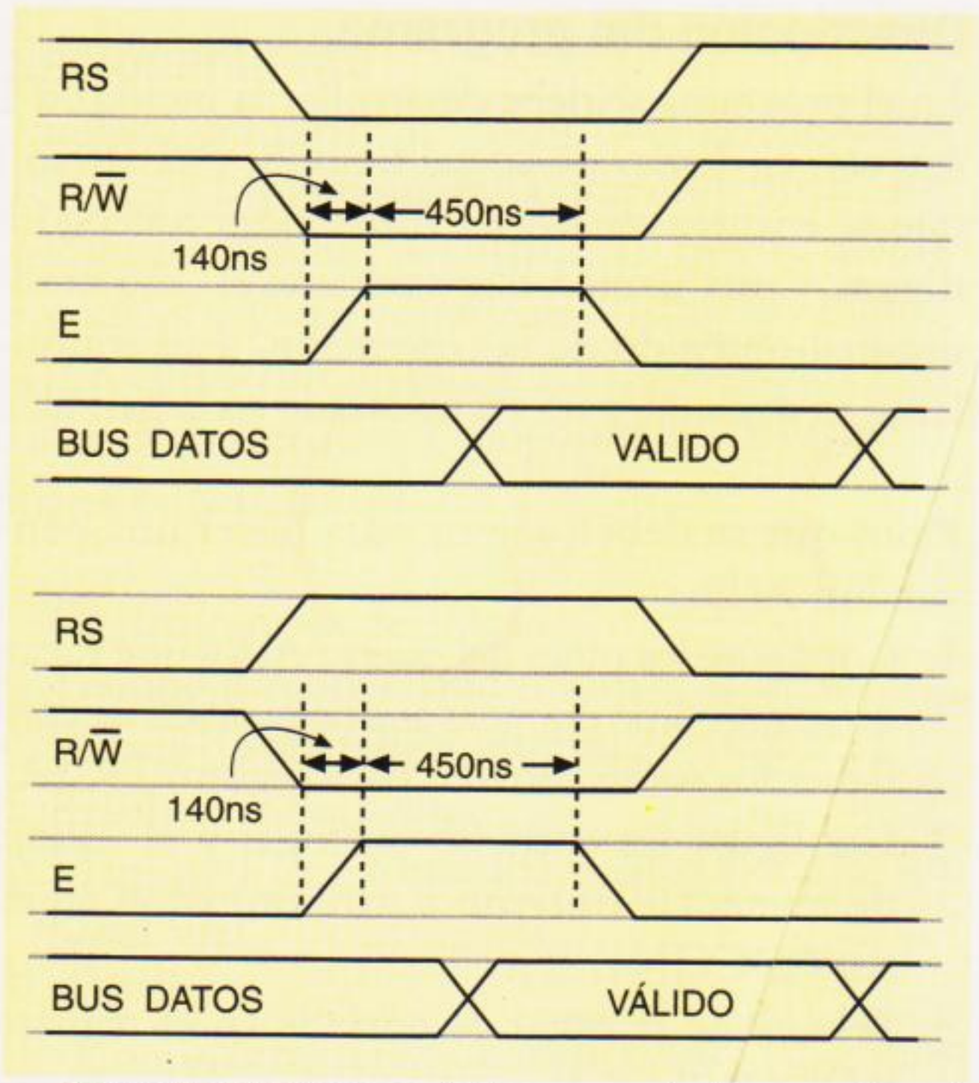


Figura 33.11 *Diagramas de tiempo para lectura y escritura de datos en un LCD*

7. Luego de iniciar el convertidor se hace un pequeño retardo (para el muestreo)
8. Se debe iniciar la conversión mediante el *bit* GO/DONE del registro ADCON0.
9. Esperar a que se termine la conversión probando el estado del *bit* GO/DONE o esperar el cambio en la bandera de interrupción, es decir, probando el *bit* ADIF del registro ADCON0.
10. El resultado se debe leer en el registro ADRES
11. Borrar el *bit* ADIF

En la **figura 33.12** observamos el diagrama de flujo del programa. Hagamos un breve análisis de sus partes.

En el primer bloque del programa (etiqueta INICIO), además de configurar los puertos, se programan los pines RA0/AN0 y RA1/AN0 como entradas análogas. El bloque que inicia con la etiqueta BEGIN se encarga de ubicar en la pantalla el mensaje de temperatura en la primera línea y el de humedad en la segunda línea.

La parte del programa que inicia con la etiqueta MIDE se encarga de configurar la velocidad del convertidor análogo a digital, de

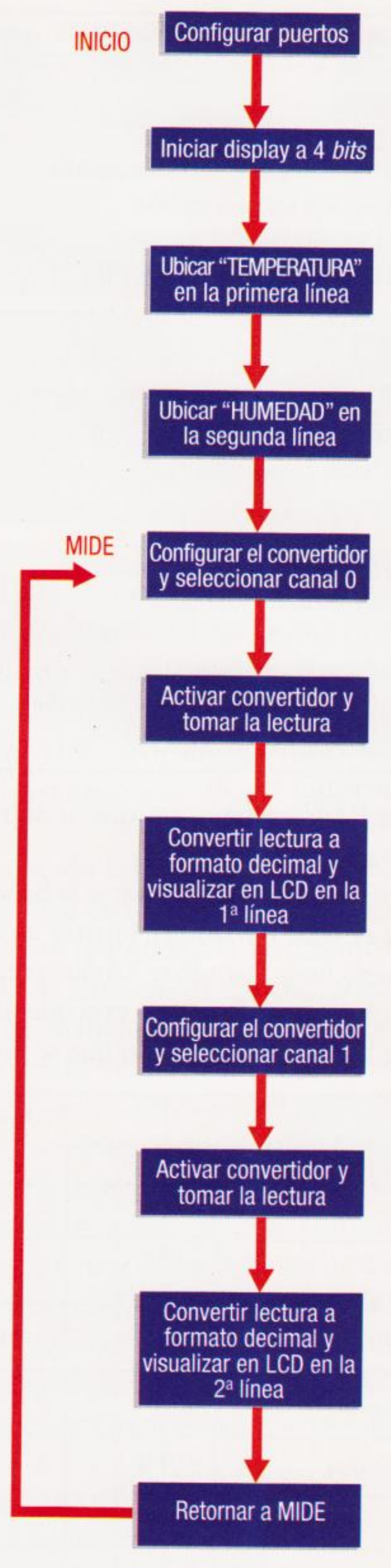


Figura 33.12 *Diagrama de flujo del programa*

poner el *bit* ADIF en "0" y de poner el *bit* ADON del registro ADCON0 en "1", antes de empezar la conversión; luego se hace un retardo. El trabajo del convertidor se inicia cuando

el *bit* ADCON0,GO se pone en "1", nuevamente se hace un pequeño retardo. Para saber si el convertidor ha terminado, se prueba el estado del *bit* llamado ADCON0,ADIF.

Una vez que se está seguro de tener el dato digital que resulta de la conversión, éste se puede leer del registro ADRES y se debe duplicar antes de enviarlo a la pantalla. Como este número se encuentra en hexadecimal, se debe utilizar una rutina que se encargue de convertirlo a su equivalente decimal. Esto se hace mediante la rutina DECIMAL, la cual devuelve el valor obtenido en los registros CENTENA, DECENA y UNIDAD. A estos datos se les suma el valor 30H para convertirlos en su equivalente ASCII antes de enviarlos a la pantalla. Este proceso se repite para cada uno de los canales, los cuales se seleccionan mediante registros; en las **figuras 33.13** y **33.14** se muestran los registros ADCON0 y ADCON1. En la **figura 33.15** se muestra el código fuente del programa.

Con este proyecto damos por finalizada la sección de proyectos de este curso. Esperamos haber cumplido con sus expectativas y haber desarrollado proyectos que les sean de mucha utilidad en sus actividades profesionales. El aspecto más sobresaliente de estos productos es que el lector mismo los puede construir debido a que los códigos fuentes se encuentran a disposición de todos nuestros lectores y alumnos.

PARA SABER MÁS:
La humedad relativa

La humedad en el aire es una medida de la cantidad de vapor de agua que éste contiene. A cualquier temperatura el aire se mantendrá absorbiendo agua hasta que toda se haya evaporado o hasta que el aire llegue a su estado de saturación.

Que el aire se encuentre saturado significa que no se llevará a cabo más evaporación debido a que está copada su capacidad de agua, a menos que se eleve la temperatura.

Humedad relativa

La humedad relativa es una forma muy apropiada de expresar la cantidad de vapor de agua contenida en un volumen de aire. Se define como el porcentaje de masa de vapor de agua presente en al aire respecto a la masa requerida para producir saturación a la misma temperatura. Cuando el aire está saturado la humedad relativa es del 100%.

Para que el ser humano se encuentre climáticamente cómodo en un sitio determinado, la humedad relativa allí debe estar en el intervalo entre 45 y 65%. Por eso la medición de la humedad es una variable que debe ser controlada tanto en edificios industriales como en sitios de trabajo.

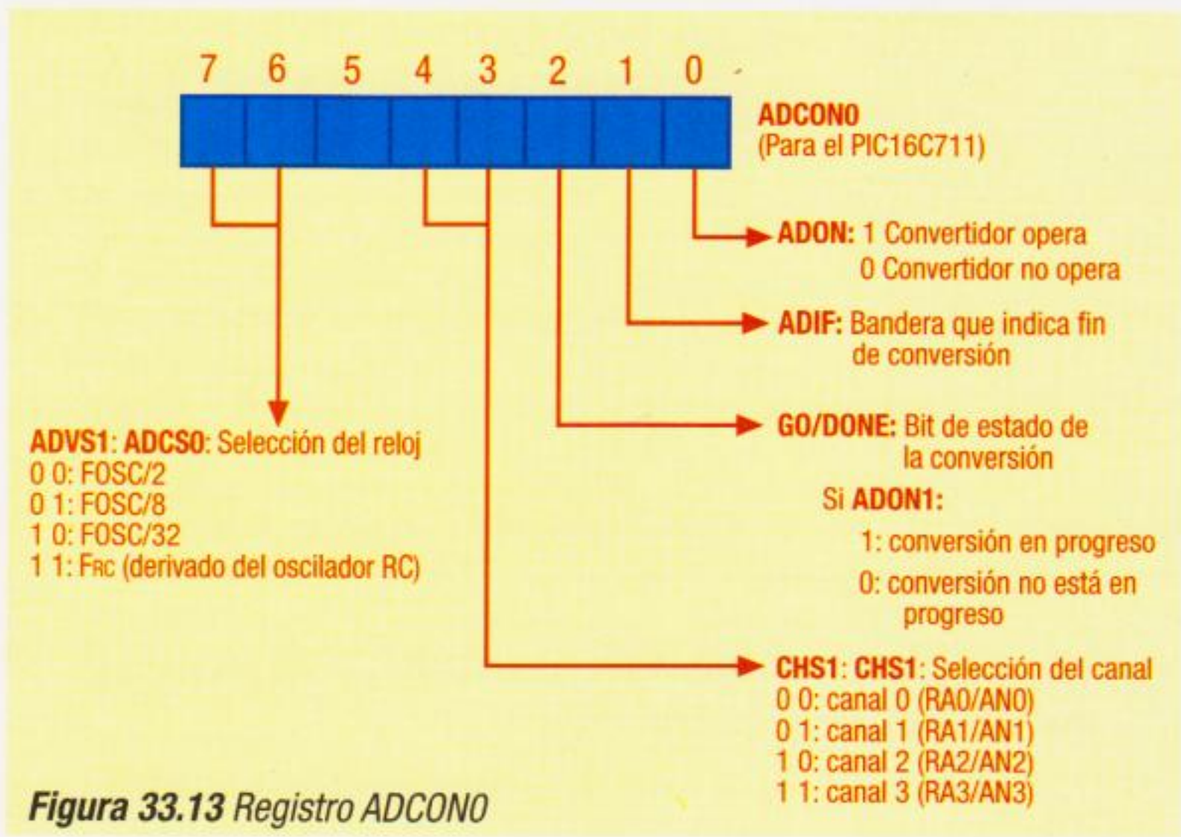
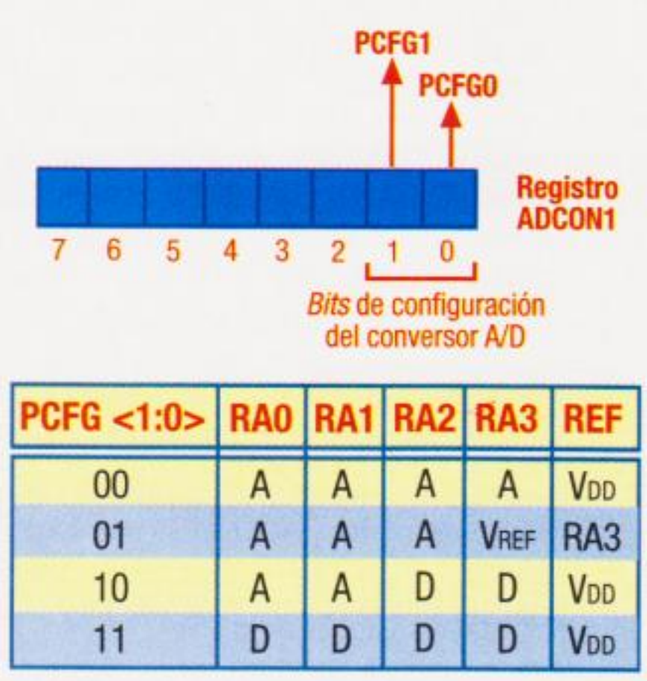


Figura 33.13 *Registro ADCON0*



PCFG <1:0>	RA0	RA1	RA2	RA3	REF
00	A	A	A	A	V_{DD}
01	A	A	A	V_{REF}	RA3
10	A	A	D	D	V_{DD}
11	D	D	D	D	V_{DD}

Figura 33.14 *Registro ADCON1*

```
;
;**********************  CURSO PRÁCTICO SOBRE MICROCONTROLADORES

; ********************** PROYECTO MEDIDOR DE HUMEDAD Y TEMPERATURA **********************************
; UTILIZA UN MÓDULO LCD DE 2 LÍNEAS 16 CARACTERES
;
;
;              ******  pic16F711  ********
;              ******  wdt = off  *******
;              ******  osc = xt   *******
;              ******   cp = on   ******
;-----------------------------------------------------------------------------
;            ******* definiciones ********
indf    equ     0h                          ;para direccionamiento indirecto
tmro    equ     1h                          ;contador de tiempo real
pc      equ     2h                          ;contador de programa
status  equ     3h                          ;registro de estados y bits de control
fsr     equ     4h                          ;seleccción de bancos de memoria y registros
ptoa    equ     5h                          ;puertos
ptob    equ     6h
adcon0  equ     8h                          ;registros del convertidor a/d
adcon1  equ     88h
adres   equ     9h
trisa   equ     85h                         ;programación de los registros
trisb   equ     86h
r0c     equ     0ch
r0d     equ     0dh
r0e     equ     0eh
unidad  equ     10h
decena  equ     11h
centena equ     12h
r14     equ     14h
r1b     equ     1bh
juan    equ    1ch
; ----------  bits especiales ---------------
                                            ;bits del registro status
rp0     equ     5h                          ;selección de página
z       equ     2h                          ;bandera de cero
c       equ     0h                          ;bandera de carry

w       equ     0h                          ;para almacenar en w
r       equ     1h                          ;para almacenar en el mismo registro
pcfg0   equ     0                           ;bits del registro adcon1
pcfg1   equ     1
adif    equ     1h                          ;bits del registro adcon0
go      equ     2h

; pines del puerto a
e       equ     1h
rs      equ     0h

;               ***********  programa principal  ************

        org     00                          ;vector de reset
        goto    inicio                      ;va a iniciar programa principal
        org     05h

retardo movlw   0ffh
        movwf   r1b
decre   decfsz  r1b,r
        goto    decre
        retlw   0

retar2  movlw   0ffh
        movwf   r14
decr2   call    retardo
        call    retardo
        decfsz  r14,r
        goto    decr2
        retlw   0

;*******************rutina que convierte binario en bcd*******************************************
decimal clrf    decena                      ;borrar registros de trabajo
        clrf    centena
        movlw   d'100'
```

```
otra    subwf   unidad,r                                  ;restar 100 al valor inicial
        btfss   status,c                                  ;verifica el carry
        goto    sum                                       ;si es cero deja de restar 100
        incf    centena                                   ;si es 1 incrementa centena
        goto    otra                                      ;volver a restar
sum     addwf   unidad                                    ;sumarle 100
        movlw   d'10'
repite  subwf   unidad                                    ;restar 10 al valor
        btfss   status,c                                  ;verifica el carry
        goto    sum1                                      ;si es 0 deja de restar
        incf    decena                                    ;si es 1 incrementa decena
        goto    repite
sum1    addwf   unidad                                    ;sumarle 10 al valor
        retlw   0                                         ;el valor de la conversion binario a decimal
                                                          ;se devuelve en los registros centena,
                                                          ;decena y unidad
;*********************rutina de control del LCD*****************************************************
control bcf     ptob,rs                                   ;esta rutina genera las señales de control
        goto    dato2                                     ;para escribir en el modulo lcd y
dato    bsf     ptob,rs                                   ;entrega el dato que va a ser mostrado en la
                                                          ;pantalla
dato2   bsf     ptob,e                                    ;utiliza la interfaz a 4 bits
        movwf   r0e
        movlw   0fh
        andwf   ptob,r
        movf    r0e,w
        andlw   0f0h
        iorwf   ptob,r
        call    retardo
        bcf     ptob,e
        call    retardo

        bsf     ptob,e
        movlw   0fh
        andwf   ptob,r
        swapf   r0e,w
        andlw   0f0h
        iorwf   ptob,r
        call    retardo
        bcf     ptob,e
        call    retardo
        retlw   0
;**********************en esta tabla se almacenan los caracteres que se muestran en el LCD*********
tabla   addwf   pc,r                                      ;mensaje que se muestra
        retlw   " "
        retlw   "H"
        retlw   "U"
        retlw   "M"
        retlw   "E"
        retlw   "D"
        retlw   "A"
        retlw   "D"
        retlw   " "
        retlw   " "
        retlw   " "
        retlw   " "
        retlw   " "
        retlw   "%"
        retlw   " "
        retlw   " "

        retlw   "T"
        retlw   "E"
        retlw   "M"
        retlw   "P"
        retlw   "E"
        retlw   "R"
        retlw   "A"
        retlw   "T"
        retlw   "U"
        retlw   "R"
        retlw   "A"
        retlw   " "
        retlw   " "
        retlw   " "
```

```
        retlw   " "
        retlw   "C"
        retlw   0
;*******************programa principal*************************************************

inicio  bsf     status,rp0          ;seleccionar página 1
        movlw   0ffh                ;configura ptoa como entradas
        movwf   trisa               ;
        bcf     adcon1,pcfg1        ;configura ra3 como voltaje de referencia
        bcf     adcon1,pcfg0        ;y ra0-ra2 como entradas análogas
        movlw   0ch                 ;configura ptob
        movwf   trisb               ;
        bcf     status,rp0          ;vuelve a página 0

begin   movlw   02h                 ;inicia display a 4 bits
        call    control
        movlw   28h                 ;display a 4 bits y 2 lineas
        call    control
        movlw   0ch                 ;activa display y desactiva cursor
        call    control
        movlw   06h                 ;selecciona el modo de desplazamiento
        call    control

blank   movlw   01h                 ;borra display y cursor a casa
        call    control
muestra movlw   0                   ;inicia contador de caracteres
        movwf   r0c
ciclo   movf    r0c,w               ;hace barrido de la tabla1
        call    tabla
        call    dato
        movlw   05fh                ;retardo entre caracteres
        movwf   r0d
reta1   call    retardo
        decfsz  r0d,r
        goto    reta1
        incf    r0c,r               ;sigue con la tabla
        movlw   10h
        subwf   r0c,w               ;pregunta si está mostrando el mensaje de la
        btfss   status,c            ;segunda línea
        goto    ciclo               ;


        movlw   10h                 ;pregunta si es la primera vez que entra
        xorwf   r0c,w               ;a la segunda línea para ir a iniciar
        btfss   status,z            ;el puntero de la ram del módulo lcd
        goto    line2               ;

linea2  movlw   0c0h                ;ubica puntero de la ram del módulo lcd
        call    control             ;en la segunda línea
line2   movlw   20h                 ;pregunta si terminó la segunda línea
        xorwf   r0c,w               ;para ir a iniciar de nuevo el mensaje o
        btfss   status,z            ;para continuar en la segunda parte del mensaje
        goto    ciclo               ;

mide    movlw   080h                ;ubica puntero de la ram del módulo lcd
        call    control             ;en la segunda linea
                                    ;se hace la conversión análogo a digital
        movlw   b'01000000'         ;configura el convertidor, selecciona canal 0
        movwf   adcon0              ;velocidad de conversión y activa el convertidor
                                    ;adif=0 antes de empezar la conversión

        bsf     adcon0,0

        movlw   0fh
        movwf   r1b
dec     decfsz  r1b,r
        goto    dec
        bsf     adcon0,go           ;empezar la conversión
        movlw   0fh
        movwf   r1b
de2     decfsz  r1b,r
        goto    de2
consu   nop                         ;
        btfss   adcon0,adif         ;pregunta fin de conversión
```

```
        goto    consu                   ;si no ha terminado sigue esperando
        bcf     adcon0,0
        bcf     adcon0,adif             ;apaga el convertidor
        movlw   b'00111001'             ;en la calibración se debe modificar este***********
        movwf   juan                    ;valor. Ver nota final del ensamblaje.   ***********
        movf    adres,w                 ;si terminó de convertir debe pasar
        subwf   juan,1
        movf    juan,w
        movwf   unidad
        addwf   unidad,r                ;duplicar el valor leido
        call    decimal                 ;convertir el dato binario a decimal
        movlw   8ah                     ;ubico puntero de ram donde va el número leido
        call    control
        movf    centena,w               ;primer dígito = centenas
        addlw   30h
        call    dato
        movlw   8bh                     ;ubico puntero de ram donde va el número leido
        call    control
        movf    decena,w                ;segundo dígito = decenas
        addlw   30h
        call    dato
        movlw   8ch                     ;ubico puntero de ram donde va el número leido
        call    control
        movf    unidad,w                ;segundo dígito = unidades
        addlw   30h
        call    dato
        call    retar2
        clrf    adres
        NOP
        NOP
        movlw   0c0h                    ;ubica puntero de la ram del módulo lcd
        call    control                 ;en la segunda línea
                                        ;se hace la conversión análogo a digital
        movlw   b'01001000'             ;configura el convertidor, selecciona canal
        movwf   adcon0                  ;velocidad de conversión y activa el convertidor
                                        ;adif=0 antes de empezar la conversión
        bsf     adcon0,0
        movlw   0fh
        movwf   r1b
dec2    decfsz  r1b,r
        goto    dec2
        bsf     adcon0,go               ;empezar la conversión
        movlw   0fh
        movwf   r1b
de22    decfsz  r1b,r
        goto    de22
consu1  nop                             ;
        btfss   adcon0,adif             ;pregunta fin de conversión
        goto    consu1                  ;si no ha terminado sigue esperando
        bcf     adcon0,0
        bcf     adcon0,adif             ;apaga el convertidor
        movf    adres,w                 ;si terminó de convertir, debe pasar
        movwf   unidad
        addwf   unidad,r                ;duplicar el valor leido
        call    decimal                 ;convertir el dato binario a decimal
        movlw   0cch                    ;ubico puntero de ram donde va el número leido
        call    control
        movf    centena,w               ;primer dígito = centenas
        addlw   30h
        call    dato
        movlw   0cdh                    ;ubico puntero de ram donde va el número leido
        call    control
        movf    decena,w                ;segundo dígito = decenas
        addlw   30h
        call    dato
        movlw   0ceh                    ;ubico puntero de ram donde va el número leido
        call    control
        movf    unidad,w                ;segundo dígito = unidades
        addlw   30h
        call    dato
        call    retar2
        clrf    adres
        goto    mide                    ;termina
        end
```

Figura 33.15 *Código fuente del programa*

Ensamblaje del proyecto

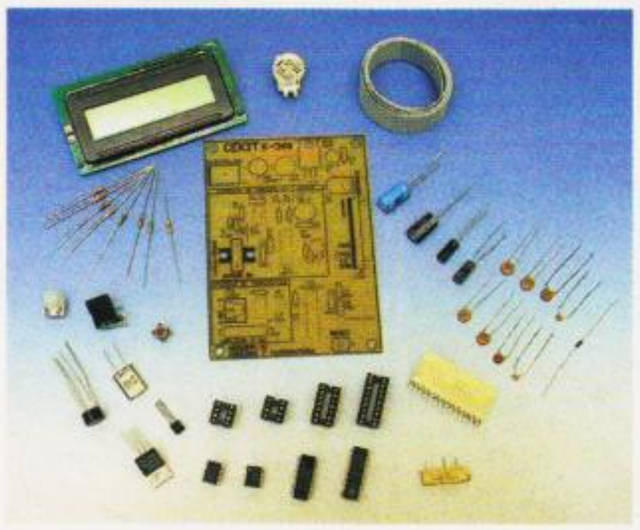
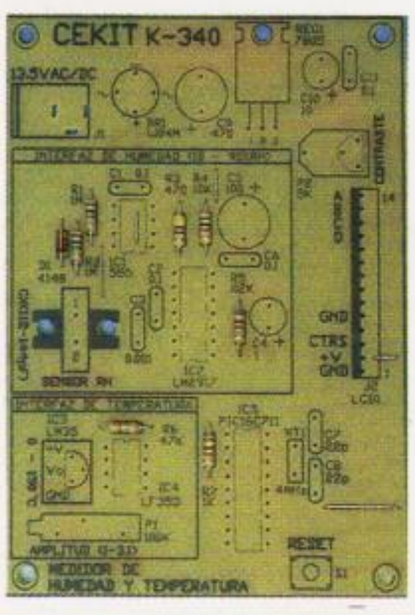


Figura 33.16 *Conjunto de componentes electrónicos y los accesorios que se requieren para ensamblar el medidor de humedad y temperatura, CEKIT K-340.*

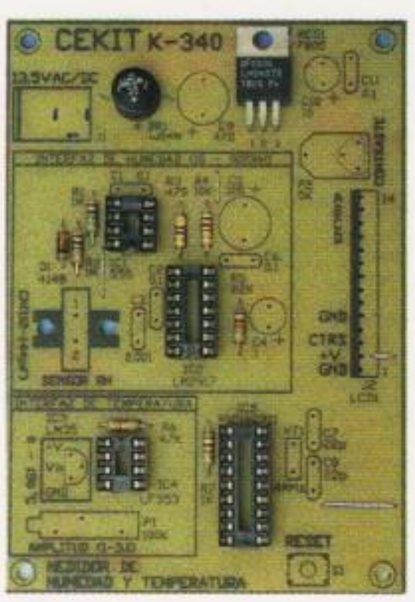
El conocimiento de la humedad y la temperatura determinan en muchos casos las condiciones de trabajo en recintos, salones, invernaderos y en algunos procesos industriales de secado y empaque . Por medio del medidor de temperatura y humedad K-340 se pueden medir y visualizar, con buena precisión estas dos magnitudes físicas. Para el ensamblaje de la tarjeta tenga en cuenta los siguientes pasos:

LISTA DE MATERIALES

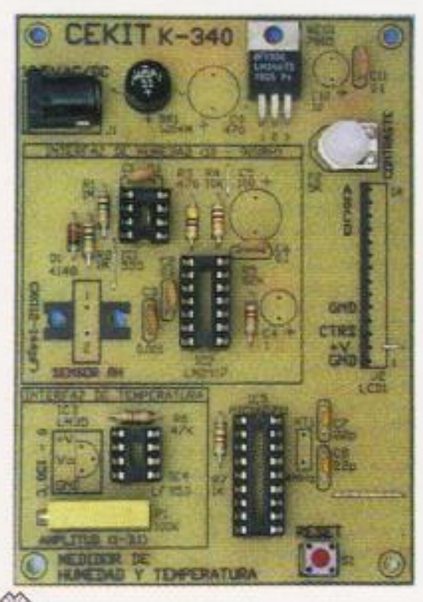
Cant.	Referencia	Descripción
1	IC1	Circuito integrado temporizador 555
1	IC2	Circuito integrado LM2917N
1	IC3	Circuito integrado sensor de temperatura LM35
1	IC4	Circuito integrado amplificador operacional LM358
1	IC5	Microcontrolador PIC16C711, programado
1	REG1	Regulador fijo de +5V, 7805
1	BR1	Puente rectificador circular de 1A, W04M
1	LCD1	Display de cristal líquido de 2 filas X 20 caracteres
1	Cx	Sensor capacitivo de humedad
4	C1,C2,C6 y C11	Condensadores cerámicos de 0,1uF - 50V
1	C3	Condensador cerámico de 0,001uF - 50V
1	C4	Condensador electrolítico de 1uF - 16V
1	C5	Condensador electrolítico de 100uF - 25V
2	C7 y C8	Condensadores cerámicos de 22pF - 50V
1	C9	Condensador electrolítico de 470uF - 25V
1	C10	Condensador electrolítico de 10uF - 25V
1	XT1	Cristal de cuarzo de 4MHz
1	D1	Diodo rectificador rápido 1N4148
2	R1 y R2	Resistencias de 1M - 1/4W
1	R3	Resistencia de 470 - 1/4W
1	R4	Resistencia de 10k - 1/4W
1	R5	Resistencia de 82k - 1/4W
1	R6	Resistencia de 47k - 1/4W
1	R7	Resistencia de 1k - 1/4W
1	P1	Trimmer resistivo de 100k
1	P2	Reostato de 5k
1	S1	Pulsador pequeño de 4 pines para circuito impreso
1	J1	Conector de alimentación CA/CC
1	J2	Conector en línea de 14 pines M/H
1	K-340	Circuito impreso CEKIT
2		Bases para circuitos integrados de 8 pines
1		Base para circuito integrado de 14 pines
1		Base para circuito integrado de 18 pines
1		Cable plano *ribbon* de 14 hilos (30cm)



Paso 1
Instale y suelde los puentes de alambre, las resistencias (R1 a R7) y el diodo 1N4148 (D1).
Figura 33.17

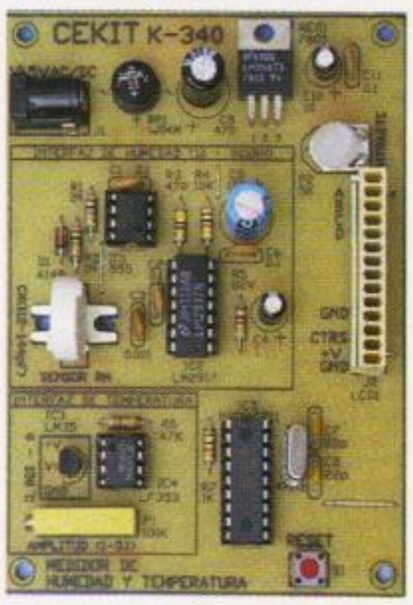


Paso 2
Luego instale y suelde las bases para los circuitos integrados, el puente rectificador circular BR1 y el regulador de voltaje integrado 7805.
Figura 33.18

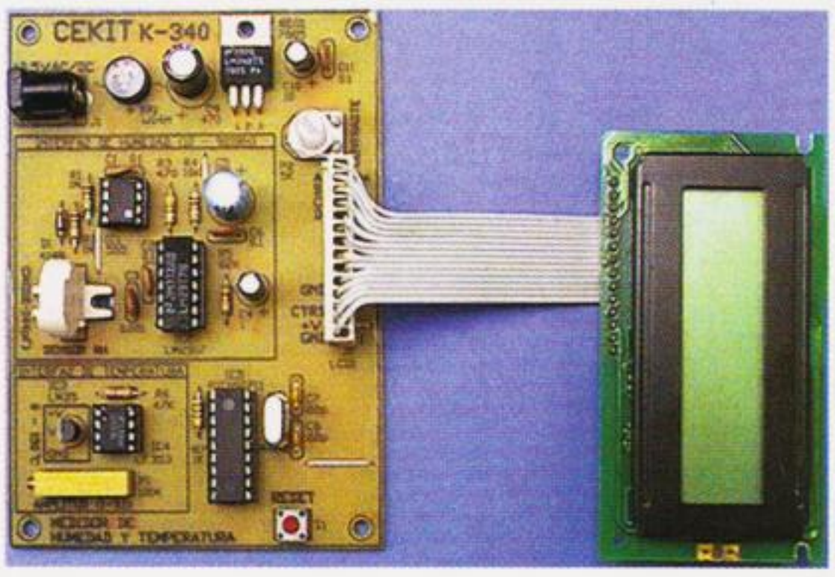


Paso 3
Instale y suelde los condensadores cerámicos, el trimmer (P1), el reóstato P2, el pulsador del reset S1 y el conector de alimentación J1.
Figura 33.19

Paso 4
Aspecto final de la tarjeta principal con sus circuitos integrados, luego de instalar y soldar los condensadores electrolíticos, el cristal de cuarzo, el conector del LCD y los sensores de humedad y temperatura.
Figura 33.20



Paso 5
Aspecto final del circuito medidor de humedad y temperatura, conectado con el display de cristal líquido.
Figura 33.21



Paso 6. Pruebas finales
*Para calibrar la temperatura leída utilice un termómetro convencional como patrón y ajuste la lectura mediante el trimmer (P1). La lectura de la humedad debe calibrarse con un higrómetro y ajustarse mediante el programa, modificando el registro **juan** (ver código del programa).*

INDICE GENERAL

INTRODUCCIÓN 1
Proyectos con microcontroladores 1
Metodología 2
Recomendación importante 4
Nota para los alumnos que se inician en el tema 4
Selección del tipo de microcontrolador 8

1 Lector binario de 4 *bits* 9
Funcionamiento 10
Memoria RAM 12
Descripción del programa 12
Ensamblaje del proyecto 16

2 Contador decimal de pulsos 17
Funcionamiento 18
Circuito de reloj 19
El decodificador de siete segmentos 20
El circuito eliminador de rebotes 20
Descripción del programa 20
La subrutina de retardo 23
Ensamblaje del proyecto 24

3 Secuenciador de 8 canales 25
Funcionamiento 26
Configuración del sistema 27
Descripción del programa 27
El *byte* de potencia 31
Ensamblaje del proyecto 32

4 Alarma para el hogar 33
Funcionamiento 34
Señalización del sistema 35
El microcontrolador 36
Arquitectura interna del 68HC908JK3 y modelo de programación 36
Mapa de memoria 37
Descripción del programa 37
La rutina de retardo 39
Ensamblaje del proyecto 40

5 Aviso luminoso con diodos LED 41
Funcionamiento 42
Efectos gráficos 43
Descripción del programa 44
La rutina *Display* Dibujo 46
Recomendaciones para el montaje 46
Ensamblaje del proyecto 48

6 Contador de eventos 49
Funcionamiento 50
Manejo de *displays* de siete segmentos 51
Descripción del programa. Concepto de macro 52
Ensamblaje del proyecto 56

7 Decodificador de teclado hexadecimal 57
Funcionamiento 58
Manejo de teclados 58
Explicación del programa 60
Ensamblaje del proyecto 64

8 Sistema de desarrollo para microcontroladores Motorola MC68HC908JK3/JK1 65
Funcionamiento 66
Pruebas iniciales de la tarjeta 68
Software de desarrollo 68
Programa ejemplo 70
Ensamblaje del proyecto 74

9 Programador "In circuit" de microcontroladores PIC 75
Características 76
Instalación del *software* y guía rápida de programación 79
Seleccionar el tipo de PIC 80
Leer PIC 80
Mostrar *buffer* 81
Programación del microcontrolador 81
Panel de pruebas del *hardware* 84
Ensamblaje del proyecto 88

10 Punta lógica 89
Funcionamiento general 90
Diagrama esquemático 91
Descripción del programa 92
Rutina de interrupción 95
Ensamblaje del proyecto 96

11 Programador de memorias seriales 97
Descripción de las memorias seriales EEPROM 98
Secuencia de operación de las memorias 99
Descripción del circuito 99
Rutinas para la programación y lectura desde la computadora (en lenguaje C) 100
Start y *stop* de la memoria 24LCXX 100
Reconocimiento en las memorias 24XX 101
Lectura y escritura de los datos en forma serial 103
Lectura de datos en memoria 24XX 103
Ensamblaje del proyecto 104

12 Generador de melodías 105
Funcionamiento 106
Descripción del programa 107
Ensamblaje del proyecto 112

13 Reloj digital con PIC 113
Descripción del circuito 114
Funcionamiento 116
Descripción del programa 117
Ensamblaje del proyecto 120

14 Microtemporizador 121
Descripción del circuito 122
Etapa de salida de potencia 122
Etapa de entradas 122
Base de tiempo 122
Despliegue de información 122
Funcionamiento 122
Descripción del programa 124
Ensamblaje del proyecto 128

15 Control de aparatos vía telefónica 129
El sistema de señalización DTMF 130
Descripción del circuito 131
El decodificador de tonos MT88L70 131
Funcionamiento 132
Operación 134
Descripción del programa 135
Ensamblaje del proyecto 136

16 Analizador lógico con PC 137
Funcionamiento 138
Descripción del circuito 139
Construcción 141
Programa 142
Pruebas finales de la tarjeta 142
Ensamblaje del proyecto 144

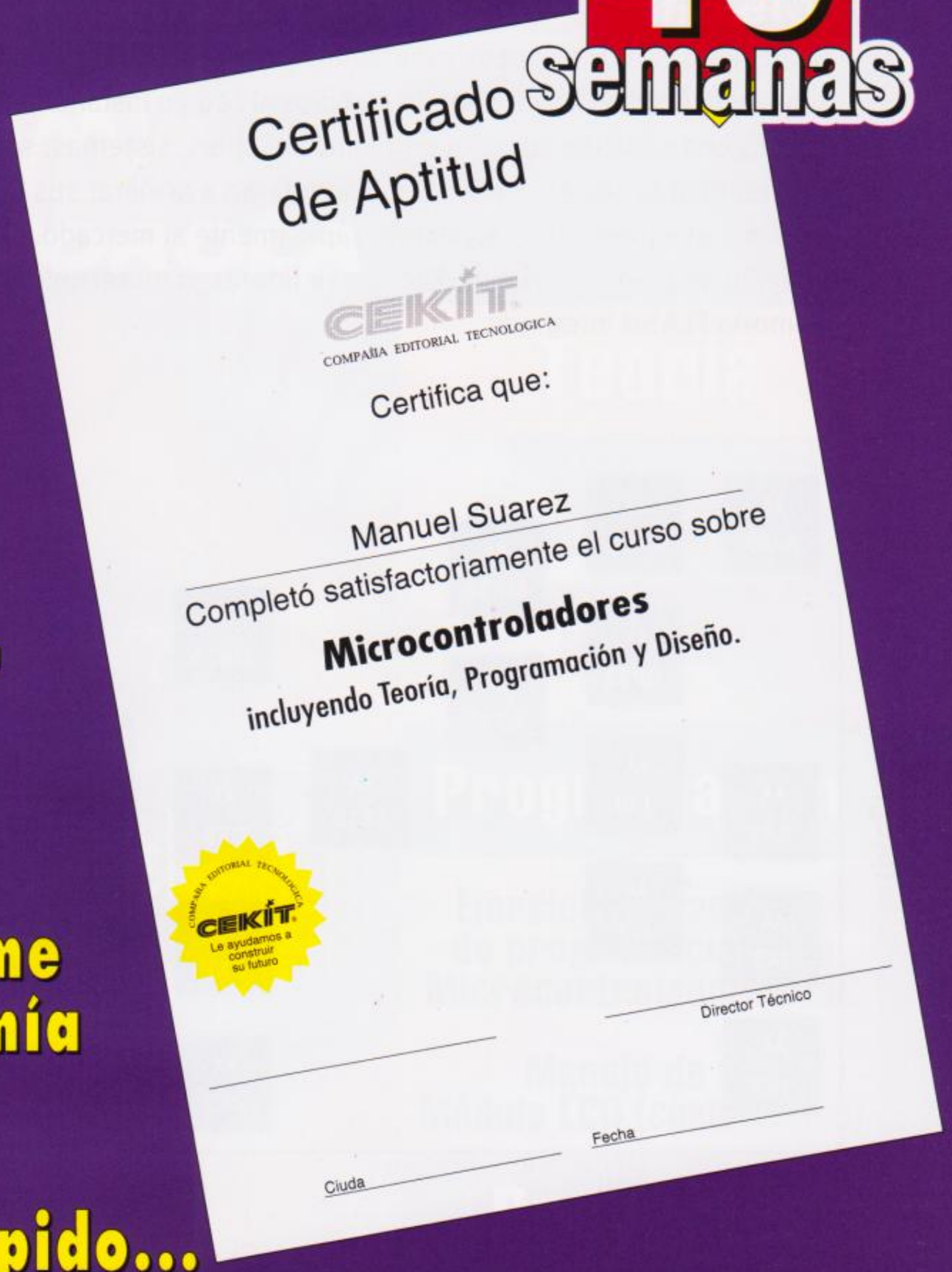
17 Control digital de temperatura ... 145
Funcionamiento ... 149
Ensamblaje ... 152
Ensamblaje del proyecto ... 155
18 Frecuencímetro digital ... 157
Funcionamiento ... 158
Lectura de los *displays* ... 158
Descripción del circuito ... 158
Descripción del programa ... 161
Pruebas de la tarjeta ... 163
Ensamblaje del proyecto ... 168
19 Teclado 4x4 con salida serial ... 169
Aplicaciones ... 170
Funcionamiento ... 171
Programa del microcontrolador ... 172
Programa de prueba con una computadora ... 175
Ensamblaje del proyecto ... 176
20 Alarma para el automóvil ... 177
Funcionamiento ... 178
Descripción del circuito ... 180
Descripción del programa ... 180
Ensamblaje del proyecto ... 184
21 Control I/O de potencia por el puerto serial ... 185
Características del programa ... 186
El sistema consta de: ... 186
Funcionamiento ... 186
Programa del microcontrolador ... 190
Programas en la computadora ... 190
Establecimiento de la comunicación ... 190
Programación de las secuencias automáticas ... 191
Operaciones ... 191
Variables ... 191
Estructura de un programa ... 194
Ensamblaje de la tarjeta electrónica ... 195
Ensamblaje del proyecto completo ... 197
Forma de conectar sensores y actuadores ... 197
Ensamblaje del proyecto ... 198
22 Alarma para motocicleta ... 201
Funcionamiento ... 202
Sensor de movimiento ... 202
Descripción del circuito ... 204
Circuito de salida con relevo ... 204
Circuito impreso del proyecto ... 205
Programa del microcontrolador ... 205
Ensamblaje del proyecto ... 208
23 Convertidor paralelo serial ... 209
Funcionamiento del circuito ... 211
Descripción del programa ... 212
Pruebas del sistema ... 214
Ensamblaje del proyecto ... 216
24 Adquisición de datos por el puerto serial ... 217
Módulo conversor análogo-digital ... 219
Módulo USART ... 221
Transmisión por la USART ... 222
Montaje del proyecto ... 222
Ensamblaje del proyecto ... 224
25 Cerradura electrónica ... 225
Funcionamiento ... 226
Circuito de potencia ... 228
Circuito lógico ... 229
Ensamblaje ... 231
Introducción de la clave ... 231
Cambio de clave ... 231
Instalación ... 232
Descripción del programa ... 233
Ensamblaje del proyecto ... 238
26 Registrador de datos en 12 *bits* ... 241
Funcionamiento ... 242
Descripción del circuito ... 242
Operación del sistema ... 245
Operación del ADS7812 ... 246
Ensamblaje del proyecto ... 248
27 Luces rítmicas ... 249
Funcionamiento ... 250
Descripción del circuito ... 250
Etapa de preamplificación ... 250
Descripción del programa ... 253
Ensamblaje del proyecto ... 256
28 Herramientas de desarrollo para BASIC Stamp I ... 257
Tarjeta para desarrollo de prototipos para BASIC Stamp I. CEKIT K-189 ... 258
Experimentando con la tarjeta de prototipos ... 258
Ensamblaje del proyecto ... 260
Ensamblaje de un ejercicio en la tarjeta de prototipos ... 261
Entrenador para BASIC Stamp I. CEKIT K-190 ... 262
29 Tarjeta para prototipos con BASIC Stamp II ... 265
Descripción general ... 266
Características ... 266
Ensamblaje ... 266
Pruebas de programación ... 268
Ensamblaje de la tarjeta de prototipos ... 268
Entrenador para microcntroladores BASIC Stamp II ... 268
Ensamblaje del proyecto ... 269
30 Minirobot con el BASIC Stamp II ... 273
Estructuras eléctrica y mecánica del robot ... 274
Los servomotores ... 274
Ensamblaje del proyecto ... 276
Ensamblaje mecánico y conexiones ... 278
Interconexión de los elementos externos a la tarjeta de control ... 280
Prueba inicial de la tarjeta ... 280
Programación del robot ... 280
31 Termómetro digital *jumbo* ... 281
Aplicaciones ... 282
Descripción ... 282
Funcionamiento ... 282
Sensor de temperatura ... 282
Circuito de control ... 285
Fuente de alimentación ... 285
Visualización ... 286
Ensamblaje ... 286
Ensamblaje del proyecto ... 288
32 Programador para microcontroladores ATMEL ... 289
Importancia de los dispositivos programadores ... 290
Programador por el puerto paralelo ... 290
Interfaz sencilla ... 290
El *software* programador ... 292
Herramientas de desarrollo adicionales ... 297
AVR Studio ... 300
La ventana del código fuente ... 300
Ventana del procesador ... 301
Ventana I/O ... 301
Ventana de memoria ... 303
Ventana de variables ... 303
Ensamblaje del proyecto ... 304
33 Medidor de humedad y temperatura ... 305
Funcionamiento ... 306
Descripción del circuito ... 306
Sensores ... 306
Etapa de procesamiento ... 308
Etapa de visualización ... 308
Descripción del programa ... 311
Ensamblaje del proyecto ... 318

Obtenga su certificado de entrenamiento en sólo 40 semanas

Al final del curso se publicará un completo cuestionario para la evaluación de sus conocimientos.

Al contestarlo correctamente, usted obtendrá un certificado de estudios expedido por CEKIT S.A.

Capacítese en Microcontroladores y forme parte activa de la economía del nuevo milenio en su profesión o su hoobie, realizando este fácil y rápido...



CURSO PRACTICO SOBRE

MICROCONTROLADORES

Otro producto con la calidad y garantía de

Es de hacer notar que el presente certificado da idea de haber cumplido con los conocimientos básicos de la teoría y práctica del curso.